НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ
імені ІГОРЯ СІКОРСЬКОГО»

Факультет інформатики та обчислювальної техніки
Кафедра автоматизованих систем обробки інформації та управління

"На правах рукопису"
УДК *004.021*

До захисту допущено
В.о. завідувача кафедри

__________ Олександр ПАВЛОВ

“ ____ ” ________ 20*20* р.

МАГІСТЕРСЬКА ДИСЕРТАЦІЯ

на здобуття ступеня магістра
за освітньо-професійною програмою
«*Інформаційні управляючі системи та технології*»
зі спеціальності 126 «*Інформаційні системи та технології*»
на тему:
«Система з підтримки дослідження оптимізаційних задач в умовах невизначеності»

Виконав:
студент VI курсу, групи **ІС-92мп**
Богданенко Микола Олексійович ___________

Керівник:
доцент, к.т.н, доцент
Жданова Олена Григорівна ___________

Консультант:
професор, д.т.н., доцент,
Жаріков Едуард В’ячеславович ___________

Рецензент:
доцент, к.т.н, доцент кафедри АУТС
Репнікова Наталія Борисівна ___________

Засвідчую, що у цій магістерській дисертації немає запозичень з праць інших авторів без відповідних посилань.
Студент _____________

Київ – 2020 року

НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ
імені ІГОРЯ СІКОРСЬКОГО»

Факультет інформатики та обчислювальної техніки
Кафедра автоматизованих систем обробки інформації та управління

Рівень вищої освіти – *другий (магістерський)*
Спеціальність – 126 «*Інформаційні системи та технології*»
Освітньо-професійна програма *«Інформаційні управляючі системи та технології»*

В.о.з авідувача кафедри

__________ Олександр ПАВЛОВ

«__» _________2020 р.

ЗАВДАННЯ
на магістерську дисертацію студенту

Богданенко Миколі Олексійовичу

1. Тема дисертації «Система з підтримки дослідження оптимізаційних задач в умовах невизначеності», науковий керівник дисертації Жданова Олена Григорівна, к.т.н, доцент кафедри АСОІУ, затверджені наказом по університету від «26» жовтня 2020 р. № 3132-с

2. Строк подання студентом дисертації “ *2* ” *12* 20*20* р.

3. Об’єкт дослідження - процес дослідження оптимізаційних задач в умовах невизначеності.
4. Перелік завдань, які потрібно розробити:
1) для обраного класу задач виконати огляд методів розв’язання;
2) розробити алгоритм вирішення задач в умовах невизначеності на основі обраного методу;
3) розробити систему із підтримки дослідження властивостей задачі лінійного програмування в умовах невизначеності, яка б дозволила проводити генерацію вхідних даних за заданими законами розподілу та виявити вплив зміни початкових умов задачі на результат (аналіз моделі на чутливість);
4) виконати аналіз отриманих результатів.

5. Орієнтовний перелік графічного (ілюстративного) матеріалу

Екранна форма головного меню системи до вибору файлу даних. Екранна форма меню вибору файлу вхідних даних. Екранна форма головного меню системи після вибору файлу даних, типів експериментів та кількості генерацій для збору статистики. Приклад формату файлу даних із параметрами для генератора задач. Приклад формату файлу даних із задачею лінійного програмування в умовах невизначеності. Гістограма впливу відсотку похибки у обмеженнях пов'язаних із несуттєвими функціоналами задачі на кількість задач, які змінили розв'язок. Схема структурна діяльності системи. Схема структурна компонентів системи. Схема структурна послідовності діяльності системи.

6. Орієнтовний перелік публікацій

Стаття та тези доповідей на науково-практичних конференціях.

7. Консультанти розділів дисертації

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв

8. Дата видачі завдання “ *1* ” *вересня* 20*20* р.

Календарний план

№ з/п	Назва етапів виконання магістерської дисертації	Строк виконання етапів магістерської дисертації	Примітка
1	*Систематизація результатів огляду літератури*	*14.09*	
2	*Пошук та аналіз існуючих методів розв'язання задачі*	*20.09*	
3	*Постановка та формалізація математичної моделі задачі*	*02.10*	
4	*Модифікація існуючих методів розв'язання задачі*	*15.10*	
5	*Розробка інформаційного та програмного забезпечення*	*01.11*	
7	*Проведення експериментальних досліджень розроблених алгоритмів*	*13.11*	
8	*Оформлення документації*	*17.11*	
9	*Подання роботи на попередній захист*	*20.11*	
10	*Подання роботи на основний захист*	*02.12*	

Студент _________ *Микола БОГДАНЕНКО*

Науковий керівник _________ *Олена ЖДАНОВА*

РЕФЕРАТ

Магістерська дисертація: 108 с., 11 рис., 7 табл., 23 джерел, 1 додаток.

Актуальність. Невизначеність має місце в усіх галузях людської діяльності і її вплив суттєвий. На даний момент існує достатньо теоретичних результатів з оптимізації в умовах невизначеності. В роботі досліджується метод розв'язання оптимізаційних задач в умовах невизначенності, запропонований Павловим О.А., на прикладі задачі лінійного програмування в умовах невизначеності. Використання цього методу дозволить підвищити якість рішеннь, які приймаються при невизначених умовах та обмеженнях, властивим багатьом системам. У зв'язку з цим актуальною є розробка системи для проведення дослідження властивостей задачі , до якої цей метод застосовується.

Зв'язок роботи з науковими програмами, планами, темами. Робота виконувалась на кафедрі автоматизованих систем обробки інформації та управління Національного технічного університету України «Київський політехнічний інститут ім. Ігоря Сікорського» в рамках теми «Ефективні методи розв'язання задач теорії розкладів» (№ ДР 0117U000919).

Мета дослідження – створення системи, яка б дозволила провести дослідження властивостей задач оптимізації певного класу в умовах невизначеності.

Для досягнення мети необхідно виконати наступні **завдання**:

– для обраного класу задач виконати огляд методів розв'язання;
– розробити алгоритм вирішення задач в умовах невизначеності на основі обраного методу;
– розробити систему із підтримки дослідження властивостей задачі лінійного програмування в умовах невизначеності, яка б дозволила проводити генерацію вхідних даних за заданими законами

розподілу та виявити вплив зміни початкових умов задачі на результат (аналіз моделі на чутливість);

– виконати аналіз отриманих результатів.

Об’єкт дослідження – процес дослідження оптимізаційних задач в умовах невизначеності.

Предмет дослідження – задача лінійного програмування в умовах невизначеності.

Методи дослідження – методи розв’язання задач оптимізації в умовах невизначеності; методи лінійного програмування.

Наукова новизна отриманих результатів

Розроблено програмну реалізацію системи для дослідження процесу оптимізації задач лінійного програмування в умовах невизначеності. На основі результатів експериментів над моделями задач лінійного програмування в умовах невизначності були виявлені нові властивості задачі.

ЗАДАЧА ЛІНІЙНОГО ПРОГРАМУВАННЯ В УМОВАХ НЕВИЗНАЧЕНОСТІ, ЧУТЛИВІСТЬ МОДЕЛІ, КОМПРОМІСНИЙ РОЗВ’ЯЗОК, СИСТЕМА ПІДТРИМКИ ДОСЛІДЖЕНЬ

ABSTRACT

Master's dissertation: 108 pages, 11 figures, 7 tables, 23 sources, 1 appendice.

Topicality. Uncertainty occurs in all areas of human activity and its impact is significant. At present, there are enough theoretical results for optimization under uncertainty. The method of solving optimization problems in conditions of uncertainty, proposed by Pavlov OA, on the example of the problem of linear programming in conditions of uncertainty, is investigated in the work. The use of this method will improve the quality of decisions made under uncertain conditions and constraints inherent in many systems. In this regard, it is important to develop a system to study the properties of the problem for which this method is used.

Connection of work with scientific programs, plans, themes. The work was performed at the Department of Automated Information Processing and Control Systems of the National Technical University of Ukraine "Kyiv Polytechnic Institute. Igor Sikorsky "in the framework of the topic" Effective methods for solving problems of schedule theory "(№ DR 0117U000919).

The purpose of the study - is to create a system that would allow to study the properties of optimization problems of a certain class in conditions of uncertainty.

To achieve this goal you must perform the following tasks:

- for the selected class of problems to review the methods of solution;
- to develop an algorithm for solving problems in conditions of uncertainty on the basis of the chosen method;
- to develop a system to support the study of the properties of the linear programming problem in conditions of uncertainty, which would allow the generation of input data according to the given distribution laws and identify the impact of changes in the initial conditions of the problem on the result
- perform an analysis of the results.

The object of research is the process of research of optimization problems in conditions of uncertainty.

The subject of research is the problem of linear programming in conditions of uncertainty.

Research methods - methods for solving optimization problems in conditions of uncertainty; methods of linear programming

Scientific novelty of the obtained results

A software implementation of the system for studying the process of optimization of linear programming problems in conditions of uncertainty has been developed. Based on the results of experiments on models of linear programming problems in conditions of uncertainty, new properties of the problem were discovered.

THE PROBLEM OF LINEAR PROGRAMMING UNDER UNCERTAINTY, MODEL SENSITIVITY, COMPROMISE SOLUTION, RESEARCH SUPPORT SYSTEM

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ

ЗЛП – задача лінійного програмування

ТЗЛП – транспортна задача лінійного програмування

ЛП – лінійне програмування

ІС – інформаційна система

ОПР – особа, що приймає рішення

ВСТУП

У процесі діяльності люди досить часто стикаються із проблемою прийняття рішення. Оскільки областей діяльності безліч, серед них можуть бути такі звичні як виробництво, організація праці, логістика, планування витрат, укладення розкладів. Залежно від того, наскільки ефективне рішення було прийняте, залежить, чи буде здатне підприємство отримувати прибуток, чи матиме можливість розвиватись далі, чи буде виконана робота, якими будуть наслідки її виконання, який ефект матимуть ці наслідки.

Прийняття рішення, зазвичай, покладено на відповідальну особу, яку обирають, залежно від того, наскільки цій особі довіряють. Зазвичай, причиною довіри слугують глибокі пізнання особи у області діяльності та достатній досвід роботи у цій області. Якщо у відповідальної особи вистачає досвіду, це дає підставу вважати, що прийняті цією людиною рішення будуть достатньо наближені до оптимальних, а отже будуть мати лиш позитивний ефект.

Нажаль, гарантованість цього сумнівна. Досвід і пізнання – це не ті величини, які можна оцінити з точки зору математики. Досвід і пізнання неможливо підрахувати, оцінити числом чи скласти рейтинг досвіду і рівня пізнання. Нехай можна оцінити кількість часу, проведеного у зайнятості в певній області діяльності, оцінити якість цієї зайнятості ніяк не можна. Отже не можемо вивести кореляцію між часом і досвідом, часом і рівнем пізнання, часом і якістю досвіду.

Максимум на який можна розраховувати – це оперувати поняттями «добрий», «задовільний», «поганий», «жахливий», коли оцінюється чийсь досвід чи знання, спираючись, у свою чергу, на суб’єктивний досвід. Виходить відносна оцінка, що базується на відносності. Якість такої оцінки так само є відносною, а отже, неможливо дати ніяких гарантій.

Із усього вищезгаданого можна зробити висновок: неможливо дати гарантію, що обрана відповідальна особа і справді має достатньо досвіду для прийняття рішення, рішення, від якого залежить прибуток і майбутнє підприємства. Оскільки вищеописаний метод не дає гарантії задовільного результату, його використання недоцільне.

Натомість, доцільним буде використання методів певної області знань - теорії прийняття рішень. У рамках цієї теорії розглядаються різноманітні методи прийняття рішень. Усі ці методи об’єднує один факт: вони так чи інакше ґрунтуються на оцінках експертів, на основі яких виконується обчислення найкращого рішення.

Чим ці методи відрізняються від прийняття рішення, як було описано вище? Методу, який було розкритиковано за низьку ефективність, не повторюваність результатів та відсутність будь-яких гарантій? Ці методи саме обчислюють, яке рішення буде найкращим, на відміну від методу, що не має жодного абсолютного обгрунтування. Вони були виведені шляхом аналізу і у якості гарантії виступає той факт, що їх математична модель детермінована, а отже, рішення можна вирахувати.

У роботі розглядається розв’язання задачі лінійного програмування в умовах невизначеності. У цій задачі, як і у класичній ЗЛП задано матрицю коефіцієнтів, обмеження, проте вектор коефіцієнтів цільової функції – невизначений. Це означає, що на момент реалізації розв’язку він може прийняти будь-який вигляд, і прийнятий розв’язок може стати неоптимальним при реалізації його у кінцевих умовах. На практиці можна виконати експертну оцінку, за якою можна побудувати ряд векторів коефіцієнтів цільової функції, які можуть стати кінцевим варіантом вектора на момент реалізації розв’язку. При розв’язанні задачі лінійного програмування в умовах невизначеності існує не один вектор, як у класичному варіанті, а обмежена лиш кількістю можливих варіантів майбутнього кількість векторів коефіцієнтів. Не можна передбачити, який із векторів стане кінцевим, не можна навіть дати оцінку ймовірності, із якою вектор стане кінцевим. Хоча, у певному сенсі, усі вектори мають однакову ймовірність стати кінцевими.

Сама задача та її розв’язок не мають прямого відношення до теорії прийняття рішень, проте, якщо проблему прийняття рішення було зведено у форму ЗЛП, отримано розв’язок цієї задачі, то на основі розв’язку може бути обране рішення, яке буде оптимальним у даних умовах.

Для знаходження оптимального рішення із теорії прийняття рішень запозичено термін «компромісний розв’язок». Це розв’язок, реалізація якого дасть найменше відхилення від оптимуму, не зважаючи, який із можливих варіантів розвитку подій станеться.

У основу цієї роботи лягли роботи [1-4]. В статтях [3-4] розглядається оптимізація транспортної задачі лінійного програмування (ТЗЛП) в умовах невизначеності. ТЗЛП – це підклас класу задач лінійного програмування, тому доцільно узагальнити розглянутий підхід до більш широкого класу, а саме на основі отриманих раніше результатів розробити алгоритм знаходження компромісного розв’язку задачі лінійного програмування в умовах невизначеності.

Як було написано у обґрунтуванні важливості цієї роботи, тема дослідження оптимізаційних задач в умовах невизначеності відносно малодосліджена, якщо порівнювати із її найближчим аналогом – алгоритмами прийняття рішень.

У основу алгоритму розв’язання задач лінійного програмування в умовах невизначеності покладена теорія, розроблена Павловим О.А і представлена в статті [1], проте, робота має на меті створити інструментарій для дослідження та провести самі дослідження у цьому напрямку. У подальшому отримані результати чи інструменти можуть стати основою для розвитку галузі, для більш глибоких досліджень. У цій роботі подано результати експериментальних досліджень, висновки із отриманих результатів та їх обґрунтування за допомогою експериментів.

1 ПОСТАНОВКА ЗАВДАННЯ МАГІСТЕРСЬКОГО ДОСЛІДЖЕННЯ

1.1 Напрям роботи

На практиці визначення найкращого варіанту дій часто виконується в умовах невизначеності. Наслідки прийнятих рішень залежать від майбутнього розвитку подій, котре може відбуватись згідно із різноманітними сценаріями.

У роботі розглядається вирішення задачі комбінаторної оптимізації в умовах невизначеності, функціонал якої у детермінованій постановці є лінійною згорткою ваг і випадкових характеристик допустимого розв’язку, а конструктивним алгоритмом її вирішення є алгоритм вирішення задачі лінійного програмування у загальному вигляді. Під невизначеністю розуміється неоднозначність коефіцієнтів функціонала оптимізації. За запропонованим у [1] підходом компромісного розв’язку по критерію мінімізації сумарно зваженого перевищення «бажаних» верхніх границь оптимальних значень часткових функціоналів буде реалізовано алгоритм розв’язання певного класу задач.

Клас задач, що розглядається у даній роботі має наступні особливості [1]:

1) критерій оптимізації, є лінійною згорткою ваг та випадкових числових характеристик допустимого розв'язку;

2) існує ефективний алгоритм розв’язання задачі у детермінованій постановці, будь-яка структурна зміна її області допустимих розв’язків (наприклад додавання лінійного обмеження) призводить до неможливості його застосування;

3) під розв’язанням задачі комбінаторної оптимізації в умовах невизначеності розуміється неоднозначність значень вагових коефіцієнтів, які входять у критерій оптимізації.

1.2 Постановка досліджуваної задачі

У якості задачі для аналізу було обрано задачі лінійного програмування. Причиною цьому слугують, по-перше, існуючі дослідження, які мають відношення до транспортних задач лінійного програмування, які, у свою чергу, є похідними від задач лінійного програмування, по-друге, відносна простота у моделюванні задач лінійного програмування комп'ютерними програмними засобами.

Усе це дає змогу проводити дослідження над цими задачами із відносною легкістю, і спрощує проведення аналізу та інтерпретацію результатів. Усе разом створює можливість проведення дослідження із отриманням значимих результатів.

Математична модель задачі лінійного програмування в умовах невизначеності має наступний вигляд:

$$f = \sum_{j=1}^{n} c_j x_j \to \min(\max),$$

$$c = \begin{cases} c^1 \\ c^2 \\ \dots \\ c^R \end{cases},$$

$$Ax = b,$$

$$\mathrm{x} \geq 0.$$

Тут вектор коефіцієнтів ЦФ c може прийняти одне із R значень, котрі задаються експертами.

1.3 Зв'язок із попередніми роботами та публікаціями

Між цим дослідженням та роботами [1]-[7], існує пряма залежність.

Роботи [5]-[7] мали відношення до теорії прийняття рішень. У роботі [5] розглядалась, у тому числі, методологія прийняття рішень при невідомих параметрах шляхом аналізу нелінійних нестаціонарних процесів.

Результати цієї роботи можуть бути застосовані як основа для створення експертних оцінок у середовищі задач в умовах невизначеності.

Шляхом створення інструментів аналізу нелінійних нестаціонарних процесів можна прийти до автоматизації процесів генерації експертних оцінок, що може бути застосовано у практичній діяльності для проведення оптимізації робочих процесів у підприємницькій діяльності.

Робота [7] була практичною реалізацією основних ідей, викладених у роботі [6]. Виконання аналізу даних дозволило створювати оцінки, за якими виконувалося прийняття рішень і подальша їх реалізація у вигляді виконання певної діяльності.

У роботі [6] приведено опис структури системи, яка розроблялась у ході цієї роботи. Робота [6] стала основою для створення системи підтримки дослідження оптимізаційних задач в умовах невизначеності у тому вигляді, у якому система існує на теперішній момент часу.

При визначенні значень майбутніх оцінок рішень можна застосовувати методи прогнозування, запропоновані в роботі [7]. Оптимізація в умовах невизначеності певною мірою співпадає із прийняттям рішень, оскільки у обох випадках існує певна міра невизначеності. Різниця полягає у тому, що оптимізація здійснюється із математичними моделями, що відносно легше, аніж прийняття рішень, де створення математичної моделі, зазвичай, являється нетривіальною задачею.

Роботи [1]-[4] лягли у основу алгоритмів, реалізованих в розробленій системі, а саме було створено основні алгоритми пошуку компромісних рішень задач в умовах невизначеності, придумано плани експериментів та досліджень.

1.4 Бажані результати

План роботи передбачає отримання певних наукових результатів. Ця робота не є виключенням, у ході роботи планується:

- визначити, чи існують у оптимізаційних задачах в умовах невизначеності параметри, які можуть суттєво вплинути на результати оптимізації в умовах невизначеності;
- визначити зв’язок між зміною параметрів моделі та результатами оптимізації в умовах невизначеності, базуючись на аналізі статистичних даних.

Висновки до розділу

У розділі розкрито основний напрям роботи, обґрунтовано вибір теми дослідження, пояснено зв’язок з попередніми науковими результатами, описано, які результати планується отримати при виконанні магістерського дослідження.

2 ОГЛЯД ТЕОРЕТИЧНИХ МАТЕРІАЛІВ ЗА ТЕМОЮ ОПТИМІЗАЦІЇ В УМОВАХ НЕВИЗНАЧЕНОСТІ

2.1 Огляд підходів та методів до розв'язання задач в умовах невизначеності

Ця робота, як і роботи [1-7], у певній мірі мають відношення до теорії прийняття рішень. Одні роботи спрямовані на отримання даних через дослідження та збір статистики, інші – на отримання теоретичних відомостей та поєднання знань у методики. Отже, оскільки усі вони мають відношення до теорії прийняття рішень, необхідно окремо згадати і про методи теорї прийняття рішень.

Теорія прийняття рішень − область знань, в якій використовуються поняття і методи економіки, статистики, математики, психології і менеджменту, це наука, що вивчає закономірності вибору людьми шляхів вирішення різного роду завдань, а також досліджує способи пошуку найкращих з можливих рішень. Ця наука охоплює теми імітаційного моделювання, багатокритеріальних рішень та управління ризиками. Це обов'язкова математична дисципліна, що забезпечує науково обґрунтований підхід до вибору найкращого варіанту (варіантів) поведінки в умовах неповної інформації про зовнішнє середовище. Неповна інформація про зовнішенє середовище – це теж форма невизначеності. У цьому магістерському дослідженні розглядається певний клас задач, а не усі можливі форми і задачі невизначеності, але зв'язок між цією роботаю та теорією прийняття рішень неможливо проігнорувати [19-23].

Теорія прийняття рішень входить до обов'язкового переліку навчальних дисциплін, які викладаються на технічних спеціальностях, пов'язаних із математичними науками, науково обґрунтований вибір альтернатив з множини можливих варіантів базується на різних математичних постановках та відповідних методах, які залежать від змісту конкретної прикладної задачі прийняття рішень [19-20].

В одних випадках задача може бути зведена до пошуку найкращої альтернативи за сукупністю критеріїв. Критеріями називають деякі оцінки альтернатив за їх привабливістю (або непривабливістю) для учасників процесу вибору, зокрема, для особи, яка приймає рішення (ОПР). В професійній діяльності вибір критеріїв часто визначається багаторічною практикою, досвідом [19-23].

У найкращому випадку, коли кожну альтернативу можна оцінити числом, тобто виставити значення критерію, порівняння альтернатив зводиться до порівняння відповідних їм чисел. Проста за постановкою задача пошуку оптимальної альтернативи за одним критерієм часто виявляється складною, оскільки метод її розв’язування визначається за специфікою множини можливих альтернатив, так і за видом самого критерію [19-23].

Задача суттєво ускладнюється при виборі альтернатив за сукупністю критеріїв, тобто, коли це задача багатокритеріальної оптимізації. Складність цієї задачі обумовлена складністю зв’зків у задачі між різними критеріями [19-23].

В реальних ситуаціях часто буває важко або неможливо дати характеристику окремої альтернативи у вигляді числового критерія або сукупності критеріїв. Але якщо розглядати альтернативу не окремо, а в парі з іншою, то знаходяться підстави сказати, яка з них краща (переважає за іншу). Можливість порівнювати альтернативи дозволяє їх впорядкувати (ранжувати) на основі мови бінарних відношень [20-23].

У теорії прийняття рішень існують методи, придатні для знаходження найкращого ріщення:

Метод аналізу ієрархій - Метод Сааті

Деякі складні задачі експертного оцінювання мають структуровану множину критеріїв. Однією з найбільш поширених структур множини критеріїв є ієрархія. У вершині ієрархії знаходиться найважливіший критерій. Деяка підмножина критеріїв утворює другий рівень ієрархії, інша підмножина – третій і т. д [20-23].

На нижньому рівні ієрархії знаходяться безпосередньо альтернативи. На рисунку 2.1 наведено загальну схему ієрархії, де $i\ j\ f$ – елементи ієрархії критеріїв, x_i – альтернативи. Верхній індекс елементів вказує рівень ієрархії, нижній – порядковий номер. Метод аналізу ієрархій реалізує декомпозицію задачі експертного оцінювання на простіші складові частини [20-23].

Унаслідок цієї декомпозиції визначається відносна значимість альтернатив за ієрархічною системою критеріїв. Відносна значимість виражається чисельно у вигляді векторів пріоритетів. Отримані таким чином значення векторів пріоритетів є оцінками у шкалі відношень і відповідають так званим жорстким оцінкам [20-23].

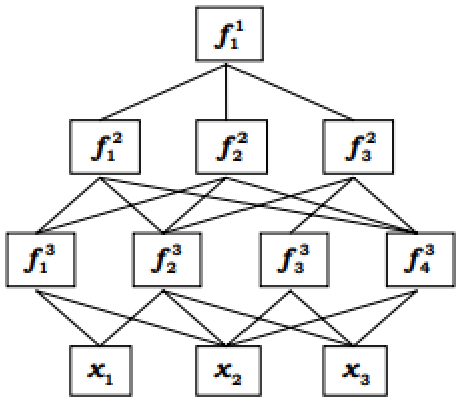


Рисунок 2.1 – Метод аналізу ієрархій

Можна виділити ряд модифікацій МАІ, які визначаються характером зв'язків між критеріями й альтернативами, розташованими на найнижчому рівні ієрархії, а також методом порівняння альтернатив. За характером зв'язків між критеріями й альтернативами визначається два типи ієрархій [21-23].

До першого типу ієрархій відносяться такі, у яких кожен критерій, що має зв'язок з альтернативами, пов'язаний з усіма розглянутими альтернативами. До другого типу - такі, у яких кожен критерій, що має зв'язок з альтернативами, пов'язаний не з усіма альтернативами [21-23].

У методі аналізу ієрархій відомі три методи порівняння альтернатив: порівняння альтернатив щодо стандартів; порівняння альтернатив копіюванням і попарне порівняння. Нижче розглядаються методологія МАІ та відмінні риси його модифікацій [23].

Метод попарного порівняння елементів ієрархії. У цій модифікації розглядається ієрархія з однаковими числом і функціональним складом альтернатив. Для установлення відносної важливості елементів ієрархії використовується шкала відношень. Ця шкала дозволяє експерту ставити у відповідність ступеням переваги одному порівнюваному об'єкту перед іншим – деяке число [23].

Матриці попарних порівнянь. Після побудови ієрархії встановлюється метод порівняння її елементів. Якщо приймається метод попарного порівняння, то будується множина матриць попарних порівнянь. Для цього в ієрархії виділяють елементи двох типів: елементи – "батьки" і елементи – "нащадки". Елементи – "нащадки" впливають на відповідні елементи вищого рівня ієрархії, які є для них "батьками". Матриці попарних порівнянь будуються для всіх елементів – "нащадків", що відносяться до відповідного елемента – "батька". Елементами – "батьками" можуть бути елементи, що належать будь-якому ієрархічному рівневі, крім останнього, на якому розташовані, як правило, альтернативи. Парні порівняння проводяться в термінах домінування одного елемента над іншим. Отримані судження виражаються в цілих числах за дев'ятибальною шкалою [23].

Ієрархічний синтез використовується для зважування власних векторів матриць попарних порівнянь альтернатив вагами критеріїв (елементів), що знаходяться в ієрархії, а також для обчислення суми по усіх відповідних зважених компонентах власних векторів нижчого рівня ієрархії. Далі розглядається алгоритм ієрархічного синтезу з урахуванням позначень, прийнятих у попередній ієрархії [23].

Оцінка однорідності ієрархії. Після розв'язання задачі ієрархічного синтезу оцінюється однорідність всієї ієрархії за допомогою підсумовування показників однорідності всіх рівнів, приведених шляхом "зважування" до першого рівня ієрархії, де знаходиться коренева вершина. Число кроків алгоритму з обчислення однорідності визначається конкретною ієрархією [23].

Агрегація думок декількох експертів. Для агрегування думок експертів береться середнє геометричне, що обчислюється за таким співвідношенням: $a_{ij}^{A} = \sqrt[n]{a_{ij}^{1} * \ldots * a_{ij}^{n}}$, де a_{ij}^{A}– агрегована оцінка елемента, що належить i-му рядку j-му стовпчику матриці попарних порівнянь; n – число матриць попарних порівнянь, кожна з яких складена одним експертом. Логічність цього критерію стає очевидною, якщо два експерти вказують при порівнянні об'єктів відповідно оцінки a і $1/a$, що при обчисленні агрегованої оцінки дає одиницю і свідчить про еквівалентність порівнюваних об'єктів. Усереднення суджень експертів може здійснюватися і на рівні власних векторів матриць попарних порівнянь [23].

Методи обробки експертної інформації

Методи обробки експертної інформації поділяються на три основні групи: статистичні методи, алгебраїчні методи й методи шкалювання [23].

Статистичні методи базуються на припущенні, що відхилення оцінок експертів від істинних значень відбувається у силу випадкових причин [23].

Суть алгебраїчних методів полягає у такому: на множині допустимих оцінок задається відстань, і результуюча оцінка визначається як така, відстань якої до оцінок експертів (за певним критерієм) мінімальна. Ідея методів шкалювання полягає у тому, що за експертною інформацією про степінь відмінності об'єктів установлюється мінімальний (або близький до мінімального) набір критеріїв та оцінок об'єктів за цими критеріями, що зумовлюють вказані експертами відмінності [23].

Методи голосування

Більшість суспільних рішень приймається на основі голосування [23].

Голосуванням обираються президенти, народні депутати, голосуванням приймаються рішення у Верховній Раді, на засіданнях Вчених рад університету і факультетів, на засіданнях кафедр, при прийнятті рішень Державною екзаменаційною комісією, у студентських колективах, у сім'ї (яку телевізійну програму дивитись) і т. п [23].

Хоча практика голосування нараховує тисячі років, фактичне його вивчення почалося близько двохсот років тому у працях французів Борда (Жан Шарль де Борд [1733–1799], фізик, математик, політик, член Французької Академії наук) і Кондорсе (Жан Антуан Ніколя де Кондорсе, [1745–1794], філософ, математик член Французької АН) [23].

Правило відносної більшості з вибуванням ("відносна більшість у два тури", "абсолютна більшість"). За подібним правилом відбуваються вибори Президента України. Якщо деякий кандидат набрав більше половини голосів, то він – переможець. Інакше до другого туру проходять два кандидати, що набрали відносну більшість голосів (тому – "відносна у два тури") [23].

Правило Кондорсе. За Кондорсе переможцем оголошується той кандидат, що "перемагає" всіх інших у попарних порівняннях. Так, у попередньому профілі: вісім виборців поставило кандидата a вище за b, дев'ять виборців поставило a нижче за b (позначимо це a:b = 8:9). Маємо b:c= 8:9, c:d = 9:8, a:c = 8:9, b:d = 5:1, a:d = 8:9. Єдиний кандидат, який "перемагає" всіх інших – це кандидат c [23].

Зауважимо, що правило Кондорсе видається вельми логічним – переможець перемагає всіх інших у єдиноборствах. Шахіст, що переміг усіх інших претендентів у мікро матчах (скажімо, із двох партій) – безумовно найкращий. Та й при формуванні індивідуальної переваги виборець попарно порівнює (свідомо чи підсвідомо) кандидатів [23].

Але, якщо кожен із кандидатів когось перемагає, а комусь програє, то у цьому випадку за визначенням переможця за Кондорсе не існує [23].

Це один із так званих "парадоксів голосування". Найпростіший випадок маємо при n = m = 3: для першого виборця a кращий за b і c, b кращий за c (позначимо це a>b>c); для другого b>c>a; для третього c>a>b [23].

Цей профіль називається "Циклом Кондорсе". Маємо – a:b = 2:1, a:c = 1:2, b:c = 2:1 (у кожного по одному виграшу і по одному програшу). Наведені вище правила застосовуються в житті, усі є досить логічними, кожне з них має свої "переваги". Але їхнє застосування до одного й того самого профілю дає абсолютно різні результати. Це ще один "парадокс голосування" [23].

Із цього можна зробити висновок, що існує два шляхи у правилах голосування. Перший – придумувати "хороші" ("логічні", "розумні") правила голосування (і сприймати результат як даність), другий – задавати "розумні" ("логічні", "хороші") умови на результат і підбирати правила, які приводять до цього результату (наприклад, "результат виборів повинен бути таким, щоб не було ображених"). Спочатку розглянемо знаменитий "парадокс Ерроу, із якого власне і почалась сучасна теорія голосування [23].

Розглянемо загальну задачу. Нехай на основі індивідуальних преваг необхідно знайти не лише "колективного" ("спільного") переможця, а й "колективний порядок". Причому, нехай, як і раніше індивідуальні переваги будуть строгими (кандидати в індивідуальних перевагах не повинні "ділити" місця), колективний же порядок може бути і нестрогим (єдиним розумним компромісом при рівноправності виборців і кандидатів у випадку переваг a>b (одного виборця) і b>a (в іншого), звичайно, буде a=b (a і b "ділять" місце)). Найпростіший метод побудови колективного порядку за даним правилом голосування є наступний – переможець виключається з профілю, для отриманого профілю знову знаходиться переможець, який займає друге місце у колективній перевазі, і т. д [23].

Теорія прийняття рішень в умовах невизначеності

Умовами невизначеності вважається ситуація, коли результати рішень, що приймаються, заздалегіть невідомі. Невизначеність підрозділяється на стохастичну (є інформація про розподіл вірогідності на безлічі результатів), поведінкову (є інформація про вплив на результати поведінки учасників), природну (є інформація тільки про можливі результати і відсутній про зв'язок між рішеннями і результатами) і апріорну (немає інформації і про можливі результати). Завдання обґрунтування рішень в умовах невизначеності усіх типів, окрім апріорної, зводиться до звуження початкової безлічі альтернатив на основі інформації, яку має в розпорядженні ОПР [23].

Якість рекомендацій для ухвалення рішень в умовах стохастичної невизначеності підвищується при обліку таких характеристик особи ОПР, як відношення до своїх виграшів і програшів, схильність до ризику. Обґрунтування рішень в умовах апріорної невизначеності можливе побудовою алгоритмів адаптивного управління [23].

Автори багатьох публікацій при вирішення задач в умовах невизначеності звертаються до використання лінійної згортки, однак, всі роботи присвячені вирішенню багатокритеріальних задач із різних областей економіки і техніки шляхом зведення їх до однокритеріального вигляду. Автори роботи [3] стверджують, що не знайшли використання лінійної згортки вагових коефіцієнтів і випадкових числових характеристик припустимого розв'язку для ефективного вирішення задачі класу в умовах невизначеності. Але, наприклад, у [15] використовується метод пошуку компромісного розв'язку двухкритеріальної задачі лінійного програмування з допомогою Lp-метрики, однак підхід у [1] принципово відрізняється вже тим, що для знаходження компромісного розв'язку досить вирішити тільки одну, а не три задачі оптимізації. Задача в приведеній постановці розглядається вперше: у статтях (наприклад, [16]) відсутнє визначення досить широкого виділеного класу задач комбінаторної оптимізації. А для вирішення різних форм невизначеності розробляються складні і трудомісткі методи стохастичної та робастної оптимізації. Але, в більшості розглянутих публікацій відмічалась надзвичайна важливість врахування динамічності та невизначеності при плануванні і складані розкладів на практиці, та необхідність врахування цієї невизначеності в моделях прийняття рішень. Тому, задачі комбінаторної оптимізації в умовах невизначеності звертають увагу спеціалістів і вчених. У [12] приведено всебічний огляд по агрегованому плануванню виробництва в умовах невизначеності, у якому зроблено висновок, що більша частина існуючих досліджень вивчає детерміновані стани планування і ігнорує властивий йому невизначений характер, що на практиці може призвести до значних помилок і неточним розв'язкам.

Усі розглянуті в цьому розділі підходи та методи не дозволяють розв'язувати задачу лінійного програмування в умовах невизначеності. Причина у тому, що математична постановка цієї задачі робить ці методи непридатними.

Магістерська дисертація є продовженням досліджень, О.А. Павлова, тут розглядаються лиш той клас задач, які можуть отримати числову характеристику до кожної альтернативи та відповідають ряду критеріїв, про які написано далі.

Висновки до розділу

У розділі розглянуто найвідоміші методи теорії прийняття рішень які можуть бути застосовані для розв’язання задач в умовах невизначеності, пояснено, що їх застосування неможливе для обраного у магістерському дослідженні класу задач.

3 ОПИС МЕТОДУ РОЗВ'ЯЗАННЯ ТА РОЗРОБКА АЛГОРИТМУ

3.1 Метод Павлова О.А. до розв'язання певного класу задач в умовах невизначеності

Згідно із матеріалами статей [1-2], цей метод працює лиш із певним класом задач. Згаданий клас задач відповідає ряду наступних критеріїв [1-2]: 1) критерій оптимізації, є лінійною згорткою ваг та випадкових числових характеристик допустимого розв'язку [1-2]; 2) існує ефективний алгоритм розв'язання задачі у детермінованій постановці, будь-яка структурна зміна її області допустимих розв'язків (наприклад додавання лінійного обмеження) призводить до неможливості його застосування [1-2]; 3) під розв'язанням задачі комбінаторної оптимізації в умовах невизначеності розуміється неоднозначність значень вагових коефіцієнтів, які входять у критерій оптимізації [1-2].

Теоретичні основи вирішення розглянутого класу задач викладено у [2,3]. Основні теоретичні положення, викладені у [1-3], полягають у наступному. Досліджуваний клас задач комбінаторної оптимізації має вигляд:

$$\min_{\sigma \epsilon \Omega} \sum_{i=1}^{s} \omega_i \, k_i(\sigma) \, , \tag{3.1}$$

ω_i - числа, $k_i(\sigma)$ – i-та випадкова числова характеристика допустимого розв'язку $\sigma(i = \overline{1,s}), \Omega$ – множина допустимих рішень.

Існує R наборів ваг $\{\omega_i, i = \overline{1,s}\}, r = \overline{1,R}$, кожен із яких може бути набором коефіцієнтів $\omega_i \dots \omega_s$ задачі (1.1) на етапі реалізації її розв'язку. Потрібно знайти допустимий розв'язок $\sigma \epsilon \Omega$, який задовольняє одному із наступних критеріїв.

Введемо позначення:

$$f_{opt}^{r} = \min_{\sigma\epsilon\Omega} \sum_{i=1}^{s} \omega_i^r k_i(\sigma)$$

$$\{\sigma_r\} = arg \min_{\sigma\epsilon\Omega} \sum_{i=1}^{s} \omega_i^r k_i(\sigma)$$

$$L_r = \sum_{m=1\, m\neq r}^{R} (\sum_{i=1}^{s} \omega_i^m k_i(\sigma) - f_{opt}^m)$$

Зауваження 1. Якщо $\{\sigma_r\}$ складається більш ніж з одного розв'язку, залишається те, на якому досягається $\min\limits_{\{\sigma_r\}} L_r$, цей розв'язок означимо як σ_r (отримання σ_r для випадку, коли Ω – кінцеве, показано в [1]).

Нехай $L_p = \min\limits_{r} L_r,$ (L_p відповідає рішенню σ_p).

Критерій 1. Необхідно знайти σ, на якому досягається

$$\min_{\sigma\epsilon\Omega} \textstyle\sum_{r=1}^{R} \left(\sum_{i=1}^{s} \omega_i^r k_i(\sigma) \; - f_{opt}^r \right), \tag{3.2}$$

Критерій 2. Знайти допустимий розв'язок, що задовольняє умові

$$\min_{\sigma\epsilon\Omega} \sum_{r=1}^{R} a_r \left(\sum_{i=1}^{s} \omega_i^r k_i(\sigma) - f_{opt}^r \right), \tag{3.3}$$

де $a_r > 0, r = \overline{1,R}$ – коефіцієнти, задані експертом.

Критерій 3. Введемо випадкову величину

$$F = \textstyle\sum_{i=1}^{s} \overline{\omega_i} k_i(\sigma) - \overline{f}_{opt},$$

де s+1-мірная дискретна випадкова величина $\overline{\omega}_1 \ldots \overline{\omega}_s, \bar{f}_{opt}$ задається таблицею:

$$\begin{Bmatrix} \omega_1^r \ldots \omega_s^r, f_{opt}^r \\ p_r > 0, r = \overline{1,R} \end{Bmatrix}.$$

Необхідно знайти розв'язок задачі

$$\operatorname*{Min}_{\sigma\epsilon\Omega} MF = \min_{\sigma\epsilon\Omega} \textstyle\sum_{r=1}^{R} p_r \left(\sum_{i=1}^{s} \omega_i^r k_i(\sigma) - f_{opt}^r \right), \tag{3.4}$$

Критерій 4. Знайти допустимий розв'язок $\sigma(\Delta_1 \ldots \Delta_R)$, у якого

$$\Delta_i \leq l_i, i = \overline{1,R}. \tag{3.5}$$

Тут $\sigma(\Delta_1 \dots \Delta_R)$ – допустимий розв'язок $\sigma\epsilon\Omega$, який має вказані відхилення від оптимумів при кожному наборі ваг:

$$\Delta_r = \sum_{i=1}^{s} \omega_i^r k_i(\sigma) - f_{opt}^r, r = \overline{1,R}, \tag{3.6}$$

Критерій 5. Для одного із набору ваг $\omega_i^r, i = \overline{1,s}, r \in \{\overline{1,R}\}$ знайти оптимальний розв'язок, якому б відповідало

$$\min_{\sigma\epsilon\{\sigma_r\}} \sum_{j=1, j\neq r}^{R} \Delta_j.$$

У випадку, якщо не вдається знайти розв'язок, що задовольняє критерію 4, в роботах [1,2] запропоновано формальна процедура знаходження компромісного розв'язку. Ця процедура базується на використанні експертних вагових коефіцієнтів обмежень $\Delta_i \leq l_i, i = \overline{1,R}$.

У магістерському дослідженні розглядаються лиш два критерії із п'яти вищезгаданих, а саме критерій 4 та критерій 5.

В роботі [1], на відміну від [13], запропоновано новий підхід до знаходження компромісного розв'язку по критерію 4 у випадку, коли конструктивний алгоритм вирішення задачі комбінаторної оптимізації в детермінованій постановці є алгоритм лінійного програмування (ЛП). Він полягає у наступному.

Знайти допустимий розв'язок $\sigma(\Delta_1 \dots \Delta_R)$, для якого досягається мінімум

$$\sum_{r=1}^{R} a_r\, y_r, \tag{3.7}$$

тут $a_r, r = \overline{1,R}$ - вагові коефіцієнти; $y_r, r = \overline{1,R}$ - величини порушення відповідних нерівностей у системі (3.5). Їх значення визначаються наступними обмеженнями:

$$\Delta_r - y_r = \sum_{i=1}^{s} \omega_i^r k_i(\sigma) - f_{opt}^r - y_r \le l_r, r = \overline{1,R}, \qquad (3.8)$$

$$y_1 \dots y_R \ge 0. \qquad (3.9)$$

Якщо $\sum_{r=1}^{R} a_r\, y_r = 0$, то отриманий розв'язок задовольняє обмеженням (3.5) і являється оптимальним по критерію 4. Якщо $\sum_{r=1}^{R} a_r\, y_r > 0$, то отриманий компромісний розв'язок являється оптимальним по критерію 5.

В роботі [13] на конкретних прикладах показано ефективність цих теоретичних положень для однопродуктової та багатопродуктової транспортної задачі, а також обгрунтовано розширеня уведеного у [2,3] класу задач комбінаторної оптимізації в умовах невизначеності. **Класична транспортна задача**

У пунктах $A_1, \dots, A_m$ виробляється деякий однорідний продукт, причому обсяг виробництва цього продукту в пункті A_i дорівнює a_i одиниць, $i = \overline{1; m}$. Зроблений у пунктах виробництва продукт повинен бути доставлений до пунктів споживання B_1,..,B_n причому обсяг споживання в пункті B_i складає b_i одиниць продукту [17].

Вважається, що транспортування готової продукції можливе з будь-якого пункту виробництва в будь-який пункт споживання і транспортні витрати, що припадають на перевезення одиниці продукту з пункту A_i в пункт B_j складають C_{ij} грошових одиниць. Задача полягає в організації такого плану перевезень, при якому сумарні транспортні витрати були б мінімальними [18].

Формально задача ставиться наступним чином. Нехай x_{ij} — кількість продукту, що перевозиться з пункту A_i в пункт B_j. Потрібно визначити сукупність з *mn* величин x_{ij}, які відповідають умовам [18]:

$$\sum_{j=1}^{n} x_{ij} = a_i,$$

$$\sum_{i=1}^{m} x_{ij} = b_i,$$

$$x_{\text{ij}} \geq 0, i = \overline{1,m}; j = \overline{1,n}.$$

Розв'язком вважається те x, для якого

$$f = \sum_{i=1}^{m} \sum_{j=1}^{n} c_{ij} x_{ij}$$

набуває найменшого значення.

3.3 Транспортна задача в умовах невизначеності

Транспортна задача лінійного програмування в умовах невизначності розглядається у роботах [3]-[4], які були відправною точкою цього магістерського дослідження.

Транспортна задача лінійного програмування в умовах невизначеності відрізняється тим, що матриця коефіцієнтів цієї задачі має декілька варіацій, кожна із варіацій представляє певний прогнозований розвиток подій, усього R варіантів. Кожнен із цих варіантів – рівноймовірний варіант розвитку подій, для транспортної задачі в умовах невизначеності математична модель приймає наступний вигляд:

$$f = \sum_{i=1}^{m} \sum_{j=1}^{n} c_{ij} x_{ij} \to min$$

$$c = \begin{cases} c^1 \\ c^2 \\ \dots \\ c^R \end{cases},$$

$$\sum_{j=1}^{n} x_{ij} = a_i,$$

$$\sum_{i=1}^{m} x_{ij} = b_i,$$

$$x_{\text{ij}} \geq 0, i = \overline{1,m}; j = \overline{1,n}.$$

Для вирішення цієї задачі розумно скористатись двома наступними критеріями для пошуку компромісного розв’язку:

Критерій 4a. Знайти допустимий розв’язок $\sigma(\Delta_1 \dots \Delta_R)$, у якого

$$\Delta_i \leq l_i, i = \overline{1, R}.$$

Тут $\sigma(\Delta_1 \dots \Delta_R)$ – допустимий розв’язок $\sigma\epsilon\Omega$, який має вказані відхилення від оптимумів при кожному наборі ваг:

$$\Delta_r = \sum_{i=1}^{s} \omega_i^r k_i(\sigma) - f_{opt}^r, r = \overline{1, R}.$$

Критерій 5a. Для одного із набору ваг $\omega_i^r, i = \overline{1, s}, r \in \{\overline{1, R}\}$ знайти оптимальний розв’язок, якому б відповідало

$$\min_{\sigma\epsilon\{\sigma_r\}} \sum_{j=1, j\neq r}^{R} \Delta_j$$

Для знаходження розв’язку цієї задачі Павловим О.А. запропоновано метод для пошуку можливого розв’язку – так званого компромісного розв’язку [1-2].

Для заходження компромісних розвязків цієї задачі за критеріями 4a і 5a створюється допоміжна задача, що має наступний вигляд:

$$\alpha_1 y_1 + \alpha_2 y_2 + \dots + \alpha_R y_R \to min, \quad (3.10)$$

$$\sum_{i=1}^{n} \sum_{j=1}^{m} c_{ij}^1 x_{ij} - f_{opt}^1 - y_1 \leq l_1, \quad (3.11)$$

$$\sum_{i=1}^{n} \sum_{j=1}^{m} c_{ij}^2 x_{ij} - f_{opt}^2 - y_2 \leq l_2, \quad (3.12)$$

$$\dots$$

$$\sum_{i=1}^{n} \sum_{j=1}^{m} c_{ij}^R x_{ij} - f_{opt}^R - y_R \leq l_R, \quad (3.13)$$

$$\sum_{j=1}^{m} x_{ij} = a_i, i = \overline{1, m}, \quad (3.14)$$

$$\sum_{j=1}^{n} x_{ij} = b_j, j = \overline{1, n}, \quad (3.15)$$

$$x_{ij} \geq 0, i = \overline{1, m}, j = \overline{1, n}, \quad (3.16)$$

$$y_i \geq 0, i = \overline{1, R}. \quad (3.17)$$

Тут $\sum_{i=1}^{n}\sum_{j=1}^{m} c_{ij}^{k}\, x_{ij} - f_{opt}^{k}$ - Δ_k, значення відхилення від оптимального часткового розв'язку, якщо застосувати компромісний розв'язок до цієї задачі.

3.3.1 Приклад вирішення транспортної задачі в умовах невизначеності

Приклад 3.1. $R = 2$; $\alpha_1 = \alpha_2 = 1$,

Знайти розв'язок, для якого виконується:

$$\Delta_i \leq l_i, i = \overline{1,2},$$

Дано:

$$a = \begin{bmatrix} 20 \\ 25 \\ 30 \\ 40 \\ 10 \\ 15 \\ 33 \end{bmatrix},\ b = \begin{bmatrix} 14 \\ 41 \\ 27 \\ 22 \\ 31 \\ 38 \end{bmatrix},\ C^1 = \begin{bmatrix} 5 & 2 & 5 & 3 & 9 & 2 \\ 3 & 4 & 2 & 5 & 2 & 8 \\ 3 & 4 & 5 & 3 & 7 & 5 \\ 8 & 5 & 9 & 9 & 1 & 4 \\ 4 & 7 & 8 & 2 & 9 & 9 \\ 2 & 4 & 6 & 5 & 3 & 2 \\ 2 & 8 & 7 & 9 & 6 & 6 \end{bmatrix},\ C^2 = \begin{bmatrix} 7 & 6 & 3 & 3 & 4 & 5 \\ 2 & 8 & 6 & 8 & 3 & 8 \\ 4 & 2 & 3 & 6 & 4 & 6 \\ 3 & 5 & 3 & 5 & 9 & 7 \\ 4 & 7 & 3 & 6 & 3 & 3 \\ 5 & 2 & 2 & 6 & 6 & 5 \\ 7 & 4 & 4 & 5 & 4 & 6 \end{bmatrix}.$$

Оптимальні розв'язки ТЗЛП1 і ТЗЛП2, а також відповідні їм значения часткові критерії приведені у таблиці 3.1.

Таблиця 3.1 – Розв'язки часткових функціоналів

Задача	Оптимальне рішення	Значення критерію	Δ_1	Δ_2
ТЗЛП1	$x^1 = \begin{bmatrix} 0 & 20 & 0 & 0 & 0 & 0 \\ 0 & 0 & 25 & 0 & 0 & 0 \\ 0 & 18 & 0 & 12 & 0 & 0 \\ 0 & 3 & 0 & 0 & 31 & 6 \\ 0 & 0 & 0 & 10 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 15 \\ 14 & 0 & 2 & 0 & 0 & 17 \end{bmatrix}$	f_{opt}^{1} =462	0	489
ТЗЛП2	$x^2 = \begin{bmatrix} 0 & 0 & 0 & 20 & 0 & 0 \\ 0 & 0 & 0 & 0 & 25 & 0 \\ 0 & 30 & 0 & 0 & 0 & 0 \\ 14 & 0 & 26 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 10 \\ 0 & 11 & 1 & 0 & 0 & 3 \\ 0 & 0 & 0 & 2 & 6 & 25 \end{bmatrix}$	f_{opt}^{2} =568	464	0

Нехай на величини Δ_1 і Δ_2 накладнено такі обмеження:

$$\Delta_1 \leq 140, \tag{3.18}$$

$$\Delta_2 \leq 120. \tag{3.19}$$

Розв'язки x^1 і x^2 не задовольняють обмеженням (3.18)-(3.19). В цьому випадку, згідно із запропонованим підходом, потрібно вирішити таку задачу ЛП:

$$y_1 + y_2 \to min$$

$$\sum_{i=1}^{7}\sum_{j=1}^{6} c_{ij}^1 x_{ij} - 462 - y_1 \leq 140,$$

$$\sum_{i=1}^{7}\sum_{j=1}^{6} c_{ij}^2 x_{ij} - 568 - y_2 \leq 120,$$

$$\sum_{j=1}^{6} x_{ij} = a_i, i = \overline{1,7},$$

$$\sum_{i=1}^{7} x_{ij} = b_i, j = \overline{1,6},$$

$$x_{ij} \geq 0, i = \overline{1,7}; j = \overline{1,6}.$$

$$y_1, y_2 \geq 0.$$

Вирішивши цю задачу, отримаємо цілочисельний розв'язок $\overline{x} = \begin{bmatrix} 0 & 0 & 0 & 12 & 0 & 8 \\ 0 & 0 & 0 & 0 & 25 & 0 \\ 0 & 22 & 8 & 0 & 0 & 0 \\ 0 & 4 & 0 & 0 & 6 & 30 \\ 0 & 0 & 0 & 10 & 0 & 0 \\ 0 & 15 & 0 & 0 & 0 & 0 \\ 14 & 0 & 19 & 0 & 0 & 0 \end{bmatrix}$, у якого $f^1(\overline{x}) = 617$, $\Delta_1 = 155$, $y_1 = 15$, $f^2(\overline{x}) = 767$, Δ_2 = 199, y_1=79, а сума величин перебільшення «бажаних» верхніх границь значень Δ_1 і Δ_2 складає $y_1 + y_2$=94. Розв'язок $\overline{x}$. Цей розв'язок являєтся оптимальним по критерію 5а - компромісним розв'язком задачі.

3.4 Класична ЗЛП

Класична задача лінійного програмування була створена ще на початку існування лінійного програмування. Можна сказати, що вона поклала основу для розвитку лінійного програмування взагалі.

З самого початку ЗЛП було створено для моделювання процесу виробництва-збуту із певними обмеженнями. Після багатьох років область застосування розширилась, проте навіть зараз використання цих задач для моделювання виробництва залишається поширеною практикою.

Задача математичного програмування вигляду:

$$\sum_{j=1}^{n} c_j x_j \to \max(min), \tag{3.20}$$

$$\sum_{j=1}^{n} a_{ij} x_j = b_i, i = \overline{1, k}, \tag{3.21}$$

$$\sum_{j=1}^{n} a_{ij} x_j \leq b_i, i = \overline{k+1, r}, \tag{3.22}$$

$$\sum_{j=1}^{n} a_{ij} x_j \geq b_i, i = \overline{r+1, m}, \tag{3.23}$$

$$x_j \geq 0, j = \overline{1, n}. \tag{3.24}$$

називається задачею лінійного програмування.

Основними допущеннями при побудові лінійних моделей, є пропорційність, адитивність, невід’ємність.

Залежно від виду обмежень розрізняють три основні форми ЗЛП.

Задача лінійного програмування вигляду (3.20) – (3.24) називається загальною ЗЛП, а ЗЛП вигляду:

$$\sum_{j=1}^{n} c_j x_j \to \max(min),$$

$$\sum_{j=1}^{n} a_{ij}\, x_j \le b_i, i = \overline{1,n},$$

$$x_j \ge 0, j = \overline{1,n}.$$

– стандартною ЗЛП. У матричному вигляді її записують так:

$$c^T x \to \max(min)\,,$$

$$Ax \le b,$$

$$x \ge 0.$$

Задача лінійного програмування вигляду

$$\sum_{j=1}^{n} c_j\, x_j \to \max(min)\,,$$

$$\sum_{j=1}^{n} a_{ij}\, x_j = b_i, i = \overline{1,n},$$

$$x_j \ge 0, j = \overline{1,n}.$$

називається канонічною ЗЛП. Її можна записати у матричному вигляді так:

$$c^T x \to \max(min)\,,$$

$$Ax = b,$$

$$x \ge 0.$$

Тут розглянуто існуючі основні методи розв’язання задач лінійного програмування. Універсальним методом розв’язання ЗЛП є симплекс-метод.

Симплекс-метод

Метод розв'язання задачі лінійного програмування, в якому здійснюється скерований рух по базисних розв’язках до знаходження оптимального розв'язку, його також називають методом поступового покращення розв’язку.

Загалом його алгоритм можна описати наступним чином:

КРОК 1. Задачу лінійного програмування звести до канонічного вигляду.

КРОК 2. Знайти допустимий базисний розв’язок.

КРОК 3. Якщо поточний ДБР є оптимальним – КІНЕЦЬ – отримали розв’язок задачі.

КРОК 4. Обрати змінну, яку буде введено в базис: ту, зростання якої призведе до покращення функціонала.

КРОК 5. Визначити елемент, що стоїть на перетині провідних рядків і стовпців.

КРОК 6. Перейти до нового базисного розв’язку.

КРОК 7. Перейти на КРОК 3.

3.5 ЗЛП в умовах невизначеності

У цьому магістерському дослідженні, на відміну від робіт [1-4], основною задачею обрано задачу лінійного програмування в умовах невизначеності. Метод О.А. Павлова, введений для певного класу задач може так само бути застосований для ЗЛП в умовах невизначеності, бо ЗЛП в умовах невизначеності має ту ж природу, отже, до неї застосовуються ті самі підходи.

Як відомо, лінійне програмування широко застосовується для моделювання виробництва із продажами. Отже, лінійне програмування в умовах невизначеності – це логічне продовження лінійного програмування із врахуванням варіативності майбутнього [6].

Як було сказано, задача в умовах невизначеності містить невизначені значення. У випадку, коли не можна визначити один вектор коефіцієнтів цільової функції, складається декілька можливих, що описують різні варіанти перебігу подій. У житті це може виглядати так: існує певне виробництво, маються ресурси, мається попит на товар, що виробляється. У класичному лінійному програмуванні розглядається один найбільш вірогідний варіант ринкових цін на товар. У задачі ж лінійного програмуванні в умовах невизначеності встановлюються декілька варіантів майбутнього [6].

Якщо прийняти, що у існує R варіантів можливого майбутнього, то математична постановка задачі матиме наступний вигляд [6]:

$$f = \sum_{j=1}^{n} c_j x_j \to \min(\max),$$

$$c = \begin{cases} c^1 \\ c^2 \\ \dots \\ c^R \end{cases},$$

$$Ax = b, \tag{3.25}$$

$$x \geq 0. \tag{3.26}$$

Без втрати загальності домовимось, що напрямком оптимізації є мінімізація цільової функції. В ідеальному випадку існує абсолютний розв’язок, який є оптимальним за будь-яким з R критеріїв [6]

$$f^r = \sum_{j=1}^{n} c_j^r x_j, r = \overline{1, R,}$$

але на практиці це малоймовірно [6].

Надалі, будемо називати *частковою* задачу [6]

$$f^r = \sum_{j=1}^{n} c_j^r x_j \to \min(\max), r = \overline{1, R},$$

$$Ax = b,$$

$$x \geq 0.$$

Якщо прийняти рішення реалізувати оптимальний розв’язок однієї з часткових задач, то скоріше за все на момент реалізації цей розв’язок не буде найкращим [6].

Для задач оптимізації в умовах невизначеності в роботах [1,2] для певного класу задач в умовах невизначеності запропонована теорія, що базується на знаходженні «компромісних розв'язків», наводиться ряд критеріїв, за якими можуть оцінюватись компромісні розв'язки, та запропоновані методи їх розв'язання. В роботах [3, 4] продемонстроване застосування розробленої теорії на прикладі транспортної задачі лінійного програмування [6].

Згідно [1-4] введемо наступні позначення для ЗЛП в умовах невизначеності [6]:

$$f_{opt}^{r} = \sum_{i=1}^{n} c_i^r x_i^r \, , r = \overline{1, R},$$

f_{opt}^{r} – оптимальне значення r-ї часткової задачі, для якої x^r – оптимальний розв'язок [6].

Нехай x – деякий допустимий розв'язок (розв'язок, який задовольняє обмеження (3.24)-(3.25)). Введемо для нього наступні величини [6]

$$\Delta_r = \sum_{i=1}^{n} c_i^r x_r - f_{opt}^{r},$$
$$y_r = \max\{0; \Delta_r - l_r\}, r = 1, \ldots, R.$$

Критерії якості компромісних розв'язків [6]

В постановці задачі лінійного програмування в умовах невизначеності, що аналізується в цій роботі, передбачається, що експертами встановлено набір величин $l_i > 0, i = \overline{1, R}$, які визначають верхні значення поступок для кожного з часткових критеріїв [6].

Критерій 4б

Знайти такий допустимий розв'язок для якого виконується [6]

$$\Delta_i \leq l_i, l_i > 0, i = \overline{1, R}.$$

Критерій 5б

Знайти такий допустимий розв'язок для якого виконується [6]

$$\sum_{r=1}^{R} \alpha_r y_r \to min$$

y_r тут – це величина перевищення Δ_r над заданим експертом обмеженням l_r [6].

Згідно [2] для знаходження компромісних розв’язків, що задовольняють критеріям 4б та 5б достатньо розв’язати наступну задачу [6].

Для розв’язання цієї задачі так само будується додаткова задача, що має наступний вигляд:

$$\alpha_1 y_1 + \alpha_2 y_2 + \cdots + \alpha_R y_R \to min, \tag{3.27}$$

$$\sum_{j=1}^{n} c_j^1 x_j - f_{opt}^1 - y_1 \le l_1, \tag{3.28}$$

$$\sum_{j=1}^{n} c_j^2 x_j - f_{opt}^2 - y_2 \le l_2, \tag{3.29}$$

$$\dots$$

$$\sum_{j=1}^{n} c_j^R x_j - f_{opt}^R - y_R \le l_R, \tag{3.30}$$

$$\sum_{j=1}^{n} a_{ij} x_j = b_i, i = \overline{1,k}, \tag{3.31}$$

$$\sum_{j=1}^{n} a_{ij} x_j \le b_i, i = \overline{k+1,r}, \tag{3.32}$$

$$\sum_{j=1}^{n} a_{ij} x_j \ge b_i, i = \overline{r+1,m}, \tag{3.33}$$

$$x_j \ge 0,\ j = \overline{1,n}, \tag{3.34}$$

$$y_i \ge 0, i = \overline{1,R}. \tag{3.35}$$

3.5.1 Приклад розв’язання ЗЛП в умовах невизначеності

Приклад 3.2: $R = 4, a_1 = \cdots = a_4 = 1.$

Знайти розв’язок, для якого виконується:

$$\Delta_i \le l_i, i = \overline{1,5}. \tag{3.36}$$

Дано:

$$A = \begin{bmatrix} 2 & 1 & 1 & 1 & 2 \\ 1 & 0 & 2 & 2 & 1 \\ 0 & 1 & 1 & 5 & 1 \\ 2 & 2 & 3 & 1 & 2 \\ 4 & 1 & 1 & 0 & 1 \end{bmatrix}, b = \begin{bmatrix} 5 \\ 15 \\ 6 \\ 12 \\ 24 \end{bmatrix}, c^1 = \begin{bmatrix} 5 \\ 1 \\ 2 \\ 4 \\ 4 \end{bmatrix}, c^2 = \begin{bmatrix} 1 \\ 2 \\ 1 \\ 5 \\ 3 \end{bmatrix},$$

$$c^3 = \begin{bmatrix} 10 \\ 2 \\ 2 \\ 6 \\ 4 \end{bmatrix}, c^4 = \begin{bmatrix} 2 \\ 11 \\ 0 \\ 4 \\ 4 \end{bmatrix}, l = \begin{bmatrix} 2 \\ 3 \\ 2 \\ 5 \end{bmatrix}.$$

Результи розв'язання наведено у таблиці 3.2:

Таблиця 3.2 – Результати розв'язання часткових задач

Задача	Оптимальний розв'язок	Значення критерію	Δ_1	Δ_2	Δ_3	Δ_4
ЗЛП1	$x_1 = \begin{bmatrix} 0 \\ 6 \\ 0 \\ 0 \\ 0 \end{bmatrix}$	$f_{opt}^1 = 6$	0	6	1	66
ЗЛП2	$x_2 = \begin{bmatrix} 0 \\ 0 \\ 4.75 \\ 0.25 \\ 0 \end{bmatrix}$	$f_{opt}^2 = 6$	4.5	0	0	1

Продовження таблиці 3.2

Задача	Оптимальний розв'язок	Значення критерію	Δ_1	Δ_2	Δ_3	Δ_4
ЗЛП3	$x_3 = \begin{bmatrix} 0 \\ 2.5 \\ 2.25 \\ 0.25 \\ 0 \end{bmatrix}$	$f_{opt}^3 = 11$	2	2	0	28.5
ЗЛП4	$x_4 = \begin{bmatrix} 0 \\ 0 \\ 6 \\ 0 \\ 0 \end{bmatrix}$	$f_{opt}^4 = 6$	6	0	1	0

Враховуючи обмеження (3.36), складемо кінцеву ЗЛП:

$$y_1 + y_2 + y_3 + y_4 \to min$$

$$\sum_{j=1}^{5} c_j^1 x_j - 6 - y_1 \le 2$$

$$\sum_{j=1}^{5} c_j^2 x_j - 6 - y_2 \le 3$$

$$\sum_{j=1}^{5} c_j^3 x_j - 11 - y_3 \le 2$$

$$\sum_{j=1}^{5} c_j^4 x_j - 6 - y_4 \le 5$$

$$Ax \le b$$

$$x_i \ge 0, i = \overline{1,5},$$

$$y_1, y_2, y_3, y_4 \ge 0.$$

В результаті обчислення даної ЗЛП, ми отримали розв’язок $x = \begin{bmatrix} 0 \\ 0.364 \\ 4.386 \\ 0.25 \\ 0 \end{bmatrix}, y = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}, f = 0$. Розв’язок оптимальний за критерієм 4б.

3.6 Загальний алгоритм розв’язання ЗЛП в умовах невизначеності та його програмна реалізація

Схема загального алгоритму розв’язання ЗЛП в умовах невизначеності має наступний вигляд:

КРОК 1: Знайти розв’язки часткових задач лінійного програмування.

КРОК 2: За формулою (3.6) обчислити значення Δ для кожного оптимального розв’язку часткових ЗЛП.

КРОК 3: Сформувати додаткову задачу лінійного програмування за методом О.А. Павлова (3.27)-(3.35).

КРОК 4: Знайти розв’язок додаткової задачі. Отриманий розв’язок і є компромісним розв’язком задачі лінійного програмування в умовах невизначеності.

Саме у такому вигляді цей алгоритм і реалізовано у системі підтримки дослідження оптимізаційних задач в умовах невизначеності – у програмному продукті цього магістерського дослідження.

Висновок до розділу

У розділі проведено огляд класичної транспортної задачі, транспортної задачі в умовах невизначеності, наведено приклад розв’язання транспортної задачі в умовах невизначеності, розглянуто класичну задачу лінійного програмування, задачу лінійного програмування в умовах невизначеності, дано приклад розв’язання задачі лінійного програмування в умовах невизначності.

Також надано огляд методів розвя’зання класичних задач лінійного програмування для двох і більше критеріїв. Пояснено зв’язок між задачами лінійного програмування та транспортними задачами, показано, що теорія, викладена у публікаціях [1-4] може бути застосована і для вирішення задач лінійного програмування в умовах невизначеності.

Також у цьому розділі розглянуто теорію прийняття рішень та методи, які в ній застосовуються. Було вказано зв’язок між теорією прийняття рішень та темою магістерської дисертації.

Результати магістерського дослідження можуть бути використані у системі прийняття рішень в умовах невизначеності на підприємствах. Важливість такого нововведення важко недооцінити, оскільки система дозволить приймати більш ефективні рішення, скоротити витрати, підвищити прибуток на підприємствах.

4 ОПИС ПРОГРАМНОГО ТА ТЕХНІЧНОГО ЗАБЕЗПЕЧЕННЯ

4.1 Призначення та функції

Призначення розробленого програмного продукту: автоматизація процесу дослідження задач в умовах невизначеності, що розглядаються у даній роботі.

Програмний продукт виконує наступні функції:

- зчитування вхідних даних для задачі в умовах невизначеності у визначеному форматі;
- розв’язання задачі;
- генерація невизначеності для вхідних даних за вказаними параметрами;
- проведення серій експериментів на заданій кількості задач певного типу з заданими параметрами, збір статистичних даних.

4.2 Середовище розробки та використані технології

Для створення програмного продукту було використано платформу .NET Framework та мову C#. В якості IDE використовується Microsoft Visual Studio 2019. Вибір мови програмування обґрунтований її гнучкістю, можливістю підключення необхідних ресурсів «на ходу», добре розвинутим набором доступних ресурсів. Розглянемо використані засоби розробки, звернувшись до офіційного сайту Microsoft.

Для вирішення задач лінійного програмування було обрано доступний у відкритому доступі набір бібліотек Google OR-Tools.

.NET Framework

Платформа .NET Framework - це інтегрований компонент Windows, який підтримує створення і виконання нового покоління програм і вебслужб XML. При розробці платформи .NET Framework враховувалися наступні її переваги:

- забезпечення середовища виконання коду, що гарантує безпечне виконання коду, включаючи код, створений невідомим або не повністю довіреним стороннім виробником;
- можливість підключення та застосування бібліотек функцій, що дозволяє суттєво скоротити час розробки шляхом включеня необхідного функціоналу без необхідності його розробки і відладки;
- розробка взаємодії на основі промислових стандартів, яка забезпечить інтеграцію коду платформи .NET Framework з будь-яким іншим кодом.
- забезпечення середовища виконання коду, що мінімізує конфлікти при розгортанні програмного забезпечення та управлінні версіями;
- виключення можливості конфлікту середовища виконання та розробленой системи;
- забезпечення узгодженого об'єктно-орієнтованого середовища програмування для локального збереження і виконання об'єктного коду, для локального виконання коду, розподіленого в Інтернеті, або для віддаленого виконання;
- забезпечення єдиних принципів роботи розробників для різних типів додатків, таких як додатки Windows і веб-застосування;
- забезпечення середовища виконання коду, що виключає проблеми з продуктивністю середовищ виконання сценаріїв або коду, що інтерпретується;
- зменшення часовитрат на створення коду по причині синтаксичної простоти та зрозумілості коду.

Google OR-Tools

Google OR-Tools - зручний набір бібліотек, який дозволяє виконувати розв’язання задач лінійного програмування. Розроблений спільними зусиллями багатьох розробників, цей інструмент дозволяє швидко та ефективно знаходити розв’язки оптимізаційних задач.

Пошук у мережі інтернет не дав ніяких інших результатів, тож цей набір бібліотек являє собою єдиний і неповторний інструмент, через що його і було обрано для використання у магістерському дослідженні.

4.3 Опис класів

Опис класів та відповідних методів наведено у таблиці 4.1.

Таблиця 4.1 – Класи, структури, об'єднанні з коротким описом.

<table>
<tr><th>Клас/структура</th><th>Атрибут/метод</th><th>Опис</th></tr>
<tr><td rowspan="2">CSVFileGenerator</td><td colspan="2">Клас, що відповідає за генерацію файлів звітів у форматі .csv. Такі звіти можна перетворити у формат Excel- файлів без втрати даних та струтури даних.</td></tr>
<tr><td>CSVFileGenerator()</td><td>Метод-конструктор, який створює екземпляр класу CSVFileGenerator. Ініціалізу змінні екземпляру, необхідні для його корректної роботи.</td></tr>
</table>

Продовження таблиці 4.1

Клас/структура	Атрибут/метод	Опис
	Void SetCanonical(LPTask task)	Метод, що дозволяє встановити задачу для порівняльного аналізу. Без виконання цього методу, неможливо створювати порівняльні звіти. У подальшому ця задача зветься еталонною.
	Void ResetTasks()	Метод, який відповідає за очистку пам'яті генератора від усіх задач, які у ньому збережені. Необхідний, якщо потрібно провести порівняння іншого кластеру задач.
	Void WriteCompareReport(string name)	Метод, що створює файл-звіт у форматі .csv з ім'ям, яке формується із переданого імені та поточної дати з часом. Звіт містить порівняння задач з еталонною, вказує різницю між ними.
	Void Add(LPTask task)	Метод для внесення нових задач у пам'ять екземпляру. У подальшому саме ці задачі порівнюються із еталонною. Необхідний для аналізу зміни границь метод.
	Void GenerateFileType1(string name)	Генерує звіт 1-го типу, у якому здійснюється вивід повної інформації про задачі.

Продовження таблиці 4.1

Клас/структура	Атрибут/метод	Опис
	Void GenerateFileType2(string name)	Генерує звіт 2-го типу, у якому здійснюється вивід часткової інформації про задачі із порівнянням результатів із еталонною.
Experiment	Клас, що відповідає за проведення експериментів з даними. Це клас-наслідник класу LPTask, має усі ті самі методи та змінні, проте доповнений функціоналом для зручнішої обробки даних та відображення.	
	LPTask(int _n, int _m, double[] _C, double[,] _A, double[] _B, double _L, char[] constrain)	Метод-конструктор, здійснює створення екземпляру класу LPTask. Ініціалізу змінні екземпляру, необхідні для його корректної роботи.
	double[] GetX()	Метод, що повретає значення вектора-розв’язку X. Цей метод необхідний для виводу результату оптимізації задачі лінійного програмування.
	bool IsMaximisationTask()	Метод, що повертає логічне значення true/false, залежно від того, який тип оптимізації установлено у задачі. Значення true виводиться, якщо у задачі виконується максимізація. Значення false виводиться, якщо у задачі виконується мінімізація.

Продовження таблиці 4.1

	void SetMaximization()	Метод, який використовується для встановлення типу оптимізації у задачі. Цей метод дозволяє виконувати максимізацію під час оптимізації задачі.
	void SetMinimization()	Метод, який використовується для встановлення типу оптимізації у задачі. Цей метод дозволяє виконувати мінімізацію під час оптимізації задачі.
	double GetResult()	Метод, який повертає оптимальне значення цільової функції задачі після оптимізаці. Виконання цього методу корректне лиш після виконання методу SolveTask(), інакше завжди повертає значення null.
	void SolveTask()	Метод, який запускає процес оптимізації задачі. Виконання цього методу корректне лиш після виконання одного з методів: SetMaximization() або SetMinimization().

Продовження таблиці 4.1

	char[] GetConstrains()	Метод, який повертає значення вектора обмежень. У цьому векторі містяться символьні елементи, які означають певний тип обмежень: ‘>’ – симовол, який відповідає обмеженню виду «≥» у задачі лінійного програмування. ‘=’ – символ, який відповідає обмеженню виду «=» у задачі лінійного програмування. ‘<’ – символ, який відповідає обмеженню виду «≤» у задачі лінійного програмування.
	static int FindStartS(List<double[]> c)	Метод, який повертає індекс початку значень додаткових обмежень, які накладають при перетворенні задачі лінійного програмування у канонічну форму задачі лінійного пограмування. Метод необхідний для коректного виводу і ствоерння звітів.
	double[] GetB()	Метод, який повретає вектор числових обмежень задачі лінійного програмування. З допомогою цих значень із часткових задач формується допоміжна задача лінійного програмування.

Продовження таблиці 4.1

	double[,] GetA()	Метод, який повретає матрицю коефіцієнтів А задачі лінійного програмування. З допомогою цих значень із часткових задач формується допоміжна задача лінійного програмування.
	double[] GetC()	Метод, який повретає вектор коефіцієнтів С задачі лінійного програмування. З допомогою цих значень із часткових задач формується допоміжна задача лінійного програмування.
	double GetL()	Метод, який повертає значення обмеження L, яке відповідає певній частковій задачі лінійного програмування. З допомогою цих значень із часткових задач формується допоміжна задача лінійного програмування.
	int GetNSize()	Метод, який повертає розмірність задачі лінійного програмування n. Це значення рівне розмірності вектора X, C.

Продовження таблиці 4.1

	int GetMSize()	Метод, який повертає розмірність задачі лінійного програмування m. Це значення рівне розмірності вектора B, L.
	void SetDescripton(string desc)	Метод, який дозволяє додати до екземпляру класу описання. Це необхідно для того, щоб промаркувати певний експеримент описом, щоб при виводі інформації про експеримент було вказано, що саме у ньому було зроблено, які вхідні параметри були у експерименту.
	string GetDescription()	Метод, який повретає описання експерименту.
	Experiment(int _n, int _m, double[] _C, double[,] _A, double[] _B, double _L, char[] constrain) : base(_n, _m, _C, _A, _B, _L, constrain)	Метод-конструктор, здійснює створення екземпляру класу LPTask. Ініціалізує змінні екземпляру, необхідні для його коректної роботи.
	void StartExperiment()	Метод, який запускає процес виконання експерименту.

Продовження таблиці 4.1

<table>
<tr><td rowspan="5">LPTask</td><td colspan="2">Це клас, що відповідає за збереження даних задачі лінійного програмування та має функціонал, що дозволяє вирішувати збережену у класі задачу. Функціонал класу дозволяє вносити у задачу зміни та давати різні вхідні параметри для досягнення бажаного користувачем результату.</td></tr>
<tr><td>LPTask(int _n, int _m, double[] _C, double[,] _A, double[] _B, double _L, char[] constrain)</td><td>Метод-конструктор, здійснює створення екземпляру класу LPTask. Ініціалізує змінні екземпляру, необхідні для його корректної роботи.</td></tr>
<tr><td>double[] GetX()</td><td>Метод, що повретає значення вектора-розв’язку X. Цей метод необхідний для виводу результату оптимізації задачі лінійного програмування.</td></tr>
<tr><td>bool IsMaximisationTask()</td><td>Метод, що повертає логічне значення true/false, залежно від того, який тип оптимізації установлено у задачі.
Значення true виводиться, якщо у задачі виконується максимізація.
Значення false виводиться, якщо у задачі виконується мінімізація.</td></tr>
<tr><td>void SetMaximization()</td><td>Метод, який використовується для встановлення типу оптимізації у задачі.
Цей метод дозволяє виконувати максимізацію під час оптимізації задачі.</td></tr>
</table>

Продовження таблиці 4.1

	void SetMinimization()	Метод, який використовується для встановлення типу оптимізації у задачі. Цей метод дозволяє виконувати мінімізацію під час оптимізації задачі.
	double GetResult()	Метод, який повертає оптимальне значення цільової функції задачі після оптимізаці. Виконання цього методу корректне лиш після виконання методу SolveTask(), інакше завжди повертає значення null.
	void SolveTask()	Метод, який запускає процес оптимізації задачі. Виконання цього методу корректне лиш після виконання одного з методів: SetMaximization() або SetMinimization().
	double[] GetB()	Метод, який повретає вектор числових обмежень задачі лінійного програмування. З допомогою цих значень із часткових задач формується допоміжна задача лінійного програмування.

Продовження таблиці 4.1

	char[] GetConstrains()	Метод, який повертає значення вектора обмежень. У цьому векторі містяться символьні елементи, які означають певний тип обмежень: ‘>’ – симовол, який відповідає обмеженню виду «≥» у задачі лінійного програмування. ‘=’ – символ, який відповідає обмеженню виду «=» у задачі лінійного програмування. ‘<’ – символ, який відповідає обмеженню виду «≤» у задачі лінійного програмування.
	static int FindStartS(List<double[]> c)	Метод, який повертає індекс початку значень додаткових обмежень, які накладають при перетворенні задачі лінійного програмування у канонічну форму задачі лінійного пограмування. Метод необхідний для корректного виводу і ствоерння звітів.
	double[,] GetA()	Метод, який повретає матрицю коефіцієнтів A задачі лінійного програмування. З допомогою цих значень із часткових задач формується допоміжна задача лінійного програмування.

Продовження таблиці 4.1

	double[] GetC()	Метод, який повретає вектор коефіцієнтів C задачі лінійного програмування. З допомогою цих значень із часткових задач формується допоміжна задача лінійного програмування.
	double GetL()	Метод, який повертає значення обмеження L, яке відповідає певній частковій задачі лінійного програмування. З допомогою цих значень із часткових задач формується допоміжна задача лінійного програмування.
	int GetNSize()	Метод, який повертає розмірність задачі лінійного програмування n. Це значення рівне розмірності вектора X, C.
	int GetMSize()	Метод, який повертає розмірність задачі лінійного програмування m. Це значення рівне розмірності вектора B, L.

Продовження таблиці 4.1

Output	Клас, що відповідає за візуальне відображення інформації у головному меню користувача.	
	void WriteLog(string text)	Статичний метод, що синхронізує доступ до головного меню, дозволяючи відображати у полі для відображення інформацію, яку видає система.
LPTaskGenerator	Клас, що здійснює генерацію задач лінійного програмування за заданими параметрами.	
	static LPTaskGenerator CreateGeneratorInstance(string path)	Метод, який відповідає за створення генератора задач із певними статичними параметрами. Наприклад, для задач в умовах невизначеності є характерною незмінність матриці ваг та обмежень. Цей метод створює генератор, що повертає задачі із однаковими вагами та обмеженнями, проте із різними векторами коефіцієнтів. У якості вхідних даних метод отримує шлях до файлу, який зчитує.

Продовження таблиці 4.1

	static LPTaskGenerator CreateGeneratorInstanceFromData(string data)	Метод, який відповідає за створення генератора задач із певними статичними параметрами. Наприклад, для задач в умовах невизначеності є характерною незмінність матриці ваг та обмежень. Цей метод створює генератор, що повертає задачі із однаковими вагами та обмеженнями, проте із різними векторами коефіцієнтів. У якості вхідних даних метод отримує необроблені дані параметрів генератора.
	LPTask GenerateTask(Random r)	Метод, який відповідає за генерацію часткових задач лінійного програмування.
	int GetCAmount()	Метод, який повертає кількість задач лінійного програмування, які мають бути згенеровані у ході роботи генератора.
	void ReadParameterFile(string data)	Метод, який виконує читання та парсинг файлу параметрів.

Продовження таблиці 4.1

ParallelTaskController	Клас, що лежить у основі цієї роботи. Здійснює контроль за розпаралеленим розв’язанням задач лінійного програмування, здійснює запуск генерації звітів, запуск порівняння результатів розв’язання, синхронізацію потоків даних, аналіз, збереження та вивід інформації.	
	ParallelTasksController(string data)	Метод-конструктор, здійснює створення екземпляру класу ParallelTasksController. Ініціалізу змінні екземпляру, необхідні для його корректної роботи. Рядковий параметр data містить у собі необроблену інформацію про задачу, яку було отримано під час читання вказаного у інтерфейсі програми файлу.
	LPTask FormSpecialTask(int _n, int _m, List<double[]> _C, double[,] _A, double[] _B, List<double> _L, List<double> results, char[] _constrains)	Метод, який відповідає за формування додаткової задачі лінійного програмування, яка використовується для пошуку компромісного розв’язку.
	void StartSolveIndividualTask(object data)	Метод, який відповідає за запуск процесу оптимізації додаткової задачі. У результатів виконання цього методу на виході отримується компромісний розв’язок задачі лінійного програмування в умовах невизначеності.

Продовження таблиці 4.1

	void StartSolvingProcess()	Метод, який відповідає за початок виконання розпаралелених обчислень та проведення експериментів. У ході роботи методу відбувається розбиття необроблених даних на дані задач, формування цих задач, ствоерння паралельних потоків, які виконують оптимізацію задач, формування додаткової задачі, яку використовують для знаходження компромісного розв’язку, та виконується проведення запрограмованих експериментів із виводом результатів та генерацією файлів звітів.
	Experiment FormExperimentalTaskType1(int _n, int _m, List<double[]> _C, double[,] _A, double[] _B, List<double> _L, List<double> results, char[] _constrains, int delta, bool positive, double chance)	Метод, що відповідає за формування задачі експерименту першого типу. У ході цього експерименту виконується аналіз моделі на чутливість за допомогою введеня випадкових величин у модель.

Продовження таблиці 4.1

	Experiment FormExperimentalTaskType2(int _n, int _m, List<double[]> _C, double[,] _A, double[] _B, List<double> _L, List<double> results, char[] _constrains, int delta, bool positive, LPTask specialTask, int CSize)	Метод, що відповідає за формування задачі експерименту другого типу. У ході цього експерименту виконується аналіз моделі на чутливість за допомогою введеня випадкових величин у обмеження пов’язані із суттевими частковими функціоналами. Пояснення про суттеві та несуттеві часткові функціонали міститься у розділі 4 РЕЗУЛЬТАТИ ЕКСПЕРИМЕНТІВ.
	Experiment FormExperimentalTaskType3(int _n, int _m, List<double[]> _C, double[,] _A, double[] _B, List<double> _L, List<double> results, char[] _constrains, int delta, bool positive, LPTask specialTask, int CSize)	Метод, що відповідає за формування задачі експерименту третього типу. У ході цього експерименту виконується аналіз моделі на чутливість за допомогою введеня випадкових величин у обмеження пов’язані із несуттевими частковими функціоналами. Пояснення про суттеві та несуттеві часткові функціонали міститься у розділі 4 РЕЗУЛЬТАТИ ЕКСПЕРИМЕНТІВ.

Продовження таблиці 4.1

ReportGenerator	Клас, що здійснює генерацію текстів звітів до розв'язаних задач лінійного програмування.	
	int GetSStart(double[] C)	Метод, який відповідає за пошук індексу першого елемента вектора розв'язку, із якого починаються додаткові значення, введені у задачу при перетворенні її у канонічну задачу лінійнного програмування.
	int GetYStart(double[] C)	Метод, який відповідає з пошук індексу першого елементу вектору розв'язку, із якого починаються додаткові значення, введені у задачу при створенні додаткової задачі лінійного програмування, створеної для пошуку компромісного розв'язку задачі лінійного програмування в умовах невизначеності. Цей метод дозволяє відділити значення вектору розв'язку від додаткових значень, тим самим покращує інформативність виведених даних.

Кінець таблиці 4.1

	string GenerateShortTaskReport(LPTask task, int index)	Метод, який відповідє за генерацію тексту короткого звіту про результати оптимізації задачі лінійного програмування. Інколи виведення додаткової інформації лиш погіршує інформативність результатів, тож виведення лиш основної інформації веде до покращення цього параметру.
	string GenerateFullTaskReport(LPTask task, int index)	Метод, який відповідає за генерацію тексту повного звіту про результати оптимізації задачі лінійного програмування. Дає повну інформацію про задачу, якщо необхідно виявити причини висновків, які спостерігаються у результаті виконання аналізу.
TaskFileReader	Клас, необхідний для читання задач із файлів із даними. Створює із даних екземпляри класів задач та передає їх у вирішувач.	
	string ReadFile(string path)	Метод, який відповідає за читання файлу завдання, або файлу із даними для генератора.
	LPTask[] CreateTasksFromData (string data)	Метод, що відповідає за генерацію задач із інформації, зчитаної із файлу завдання.

4.4 Діаграма послідовності роботи системи

На діаграмі послідовності зображено послідовність виклику функцій та виконання внутрішніх процесів системи під час взаємодії із користувачем. Зображення діаграми наведено у додатку А під назвою «Схема структурна послідовності системи».

На діаграмі зображено процеси вибору файлу даних із задачею, вирішення цієї задачі у паралельному режимі, вибір файлу даних для генератора, генерація задач та виконання експериментів над згенерованими задачами. Процес звершення роботи програми.

4.5 Діаграма компонентів системи

На діаграмі компонентів системи зображено із яких основних компонентів складається система. Зображення діаграми компонентів наведено у Додатку А під назвою «Схема структурна компонентів системи».

На схемі є компонент «SolverApplication» - цей компонент представляє собою програму уцілому, та компонент Google OR-Tools – цей компонент представляє собою зовнішню бібліотеку, яку використано для виконання обчислень у рамках роботи системи.

Усередині компонента «SolverApplication» є два компоненти із схожими назвами «ParallelTaskController(solver)» та «ParallelTaskController(experimenter)». Ці компоненти представляють собою один компонент, який працює у різних режимах. У режимі вирішувача компонент виконує розв’язання задач, а у режимі експериментатора – виконує проведення експериментів.

У кожному із цих компонентів існують ще менші компоненти, які мають назви, що повторюються: «LPTask» та «Experiment». Ці компоненти представляють собою паралельні процеси, у яких виконується обчислення часткових задач лінійного програмування та проведення експериментів.

4.6 Діаграма діяльності системи

На діаграмі діяльності системи зображено послідовність дій, які виконуються під час роботи системи. Зображення діаграми представлено у Додатку А під назвою «Схема структурна діяльності системи».

Висновки до розділу

В цьому розділі розкрито використані у ході магістерського дослідження технології, призначення та функції програмного продукту, розглянуто класи та методи, які використані у програмному продукті магістерського дослідження, пояснено, які функції вони виконують, надано детальні пояснення до структурних схем магістерського дослідження.

5 РЕЗУЛЬТАТИ ЕКСПЕРИМЕНТАЛЬНИХ ДОСЛІДЖЕНЬ

5.1. Опис змісту розділу

У цьому розділі викладено опис експериментів, які було проведено, теоретичне обґрунтування, статистичні дані та надано отримані у ході експериментів результати.

Для виконання усіх викладених нижче експериментів застосовувалось програмне забезпечення, яке було розроблено для дослідження властивостей оптимізаційних задач в умовах невизначеності.

При проведені експериментів встановлено, що результати, отримані із задачами створеними генератором мають суттєві відмінності від результатів задач, які були згенеровані людиною.

У кожному параграфі надано результати для задач, згенерованих експертом для порівняння.

5.2. Типи експериментів

Усього було проведено 3 типи експериментів, при чому 2 із них були необхідні для підтвердження гіпотези, яка виникла після проведення 1-го типу. Детально про типи і зміст експериментів написано у наступних параграфах, там же надано результати обчислень, статистичні дані, висновки.

5.3. Перший тип експериментів

Перший тип експериментів мав на меті виявлення міри впливу змін моделі задачі на отриманий в результаті пошуку рішення результат.

Для цього випадкові вектори коефіцієнтів С задачі лінійного програмування змінювали від 1% до 10% відносно початкового значення. Наприклад, у першому вектор С із десяти, до кожного із значень вектора додавали або віднімали по 1-10% від абсолютного значення. У ході експерименту зміна значень була не разова, а була поступовою, і на кожному етапі фіксувались зміни у розв’язку та у результаті його реалізації.

На рисунку 5.1 та рисунку 5.2 наведено вивід розв’язку задачі лінійного програмування в умовах невизначеності.

Рядки, підписані як C:, X|Y: - мають вивід значень векторів коефіцієнтів C для додаткової задачі та суміщеного вектора розв’язку X|Y. Як відомо із розділу 3, для вирішення задачі лінійного програмування в умовах невизначеності, за створеним алгоритмом, формується додаткова задача, у якій значення X - вектора розв’язку фігурують лиш у нерівностях-обмеженнях, а додаткові значення вектора Y – не тільки у нерівностях-обмеженнях, але й у ЦФ задачі.

Рядки нижче відповідають за обмеження цієї задачі.

```
LP Task #0 report

C*x->min
0,000*0,000+0,000*2,800+0,000*1,714+0,000*1,714+0,000*1,514+0,000*0,000+1,000*0,000+1,000*0,971+1,000*0,000+1,000*2,514=3,486

C:      0,000   0,000   0,000   0,000   0,000   0,000   |       1,000   1,000   1,000   1,000
X|Y:    0,000   2,800   1,714   1,714   1,514   0,000   |       0,000   0,971   0,000   2,514


        Matrix A |        B(constrains)

3,000*0,000   3,000*2,800   2,000*1,714   2,000*1,714   2,000*1,514   3,000*0,000   -1,000*0,000   0,000*0,971    0,000*0,000    0,000*2,514     <=   18,286   (18,286)
2,000*0,000   2,000*2,800   3,000*1,714   1,000*1,714   1,000*1,514   3,000*0,000   0,000*0,000    -1,000*0,971   0,000*0,000    0,000*2,514     <=   13,000   (13,000)
3,000*0,000   1,000*2,800   1,000*1,714   1,000*1,714   1,000*1,514   3,000*0,000   0,000*0,000    0,000*0,971    -1,000*0,000   0,000*2,514     <=   9,600    (7,743)
3,000*0,000   3,000*2,800   2,000*1,714   3,000*1,714   3,000*1,514   2,000*0,000   0,000*0,000    0,000*0,971    0,000*0,000    -1,000*2,514    <=   19,000   (19,000)
3,000*0,000   3,000*2,800   2,000*1,714   3,000*1,714   2,000*1,514   3,000*0,000   0,000*0,000    0,000*0,971    0,000*0,000    0,000*2,514     >=   20,000   (20,000)
2,000*0,000   2,000*2,800   3,000*1,714   3,000*1,714   2,000*1,514   2,000*0,000   0,000*0,000    0,000*0,971    0,000*0,000    0,000*2,514     >=   16,000   (18,914)
3,000*0,000   3,000*2,800   3,000*1,714   2,000*1,714   2,000*1,514   2,000*0,000   0,000*0,000    0,000*0,971    0,000*0,000    0,000*2,514     >=   20,000   (20,000)
2,000*0,000   3,000*2,800   2,000*1,714   3,000*1,714   3,000*1,514   3,000*0,000   0,000*0,000    0,000*0,971    0,000*0,000    0,000*2,514     >=   13,000   (21,514)
3,000*0,000   3,000*2,800   3,000*1,714   2,000*1,714   2,000*1,514   2,000*0,000   0,000*0,000    0,000*0,971    0,000*0,000    0,000*2,514     >=   15,000   (20,000)
2,000*0,000   2,000*2,800   2,000*1,714   2,000*1,714   3,000*1,514   2,000*0,000   0,000*0,000    0,000*0,971    0,000*0,000    0,000*2,514     >=   17,000   (17,000)
2,000*0,000   2,000*2,800   3,000*1,714   2,000*1,714   3,000*1,514   2,000*0,000   0,000*0,000    0,000*0,971    0,000*0,000    0,000*2,514     >=   18,000   (18,714)
```

Рисунок 5.1 – Приклад виводу системи: вивід результатів пошуку компромісного розв’язку.

```
Experiment 1-a-1: Delta 1%, positive


LP Task #0 report

C*x->min
0,000*0,000+0,000*2,547+0,000*1,895+0,000*1,895+0,000*1,442+0,000*0,000+1,000*0,000+1,000*1,257+1,000*0,000+1,000*2,656=3,914

C:     0,000   0,000   0,000   0,000   0,000   0,000   |      1,000   1,000   1,000   1,000
X|Y:   0,000   2,547   1,895   1,895   1,442   0,000   |      0,000   1,257   0,000   2,656


       Matrix A |       B(constrains)

3,030*0,000   3,030*2,547   2,020*1,895   2,020*1,895   2,020*1,442   3,030*0,000   -1,000*0,000   0,000*1,257    0,000*0,000    0,000*2,656     <=   18,286   (18,286)
2,020*0,000   2,020*2,547   3,030*1,895   1,010*1,895   1,010*1,442   3,030*0,000   0,000*0,000    -1,000*1,257   0,000*0,000    0,000*2,656     <=   13,000   (13,000)
3,030*0,000   1,010*2,547   1,010*1,895   1,010*1,895   1,010*1,442   3,030*0,000   0,000*0,000    0,000*1,257    -1,000*0,000   0,000*2,656     <=   9,600    (7,857)
3,030*0,000   3,030*2,547   2,020*1,895   3,030*1,895   3,030*1,442   2,020*0,000   0,000*0,000    0,000*1,257    0,000*0,000    -1,000*2,656    <=   19,000   (19,000)
3,000*0,000   3,000*2,547   2,000*1,895   3,000*1,895   2,000*1,442   3,000*0,000   0,000*0,000    0,000*1,257    0,000*0,000    0,000*2,656     >=   20,000   (20,000)
2,000*0,000   2,000*2,547   3,000*1,895   3,000*1,895   2,000*1,442   2,000*0,000   0,000*0,000    0,000*1,257    0,000*0,000    0,000*2,656     >=   16,000   (19,349)
3,000*0,000   3,000*2,547   3,000*1,895   2,000*1,895   2,000*1,442   2,000*0,000   0,000*0,000    0,000*1,257    0,000*0,000    0,000*2,656     >=   20,000   (20,000)
2,000*0,000   3,000*2,547   2,000*1,895   3,000*1,895   3,000*1,442   3,000*0,000   0,000*0,000    0,000*1,257    0,000*0,000    0,000*2,656     >=   13,000   (21,442)
3,000*0,000   3,000*2,547   3,000*1,895   2,000*1,895   2,000*1,442   2,000*0,000   0,000*0,000    0,000*1,257    0,000*0,000    0,000*2,656     >=   15,000   (20,000)
2,000*0,000   2,000*2,547   2,000*1,895   2,000*1,895   3,000*1,442   2,000*0,000   0,000*0,000    0,000*1,257    0,000*0,000    0,000*2,656     >=   17,000   (17,000)
2,000*0,000   2,000*2,547   3,000*1,895   2,000*1,895   3,000*1,442   2,000*0,000   0,000*0,000    0,000*1,257    0,000*0,000    0,000*2,656     >=   18,000   (18,895)
```

Рисунок 5.2 – Приклад виводу системи: вивід результатів пошуку компромісного розв’язку при внесенні 1% похибки у випадкові вектори обмежень, які формувались із часткових ЗЛП

У ході цього типу експериментів було помічено, що внесення змін у випадкові обмеження, створені із функціоналів часткових задач – векторів коефіцієнтів C, у різній мірі впливають на результати розв’язання: коли внесення змін у одні обмеження моментально змінювало розв’язок, зміни у інших або зовсім, або мало впливали на результати розв’язання.

Із цього було зроблено висновок, що, ймовірно, у задачах в умовах невизначеності існують обмеження, пов’язані із суттєвими та несуттєвими частковими функціоналами, і висунуто гіпотезу про існування таких часткових функціоналів у задачі лінійного програмування в умовах невизначеності.

Внесення змін у обмеження пов’язані із суттєвими частковими функціоналами моментально змінює результати, коли внесення змін у обмеження, пов’язані із несуттєвими функціоналами або не має ніякого впливу, або цей вплив достатньо малий.

Суттєвими було названо ті функціонали, обмеження яких при реалізації оптимального розв’язку мали значення достатньо близьке до граничного, а несуттєвими – усі інші.

Наприклад, на рисунку 5.2 та рисунку 5.3 такими обмеженнями є 1,2,4 і 1,4,6,8 відповідно.

LP Task #0 report

C*x->min

0,000*0,084+0,000*0,137+0,000*3,432+0,000*0,400+0,000*0,432+1,000*2,747+1,000*0,000+1,000*0,000+1,000*0,000+1,000*0,000+1,000*13,989+1,000*0,000+1,000*0,000=16,737

C:	0,000	0,000	0,000	0,000	0,000	|	1,000	1,000	1,000	1,000	1,000	1,000	1,000	1,000
X|Y:	0,084	0,137	3,432	0,400	0,432	|	2,747	0,000	0,000	0,000	0,000	13,989	0,000	0,000

Matrix A | B(constrains)

5,000*0,084	1,000*0,137	2,000*3,432	4,000*0,400	4,000*0,432	-1,000*2,747	0,000*0,000	0,000*0,000	0,000*0,000	0,000*0,000	0,000*13,989	0,000*0,000	0,000*0,000	<=	8,000	(8,000)
1,000*0,084	2,000*0,137	1,000*3,432	5,000*0,400	3,000*0,432	0,000*2,747	-1,000*0,000	0,000*0,000	0,000*0,000	0,000*0,000	0,000*13,989	0,000*0,000	0,000*0,000	<=	9,000	(7,084)
10,000*0,084	2,000*0,137	2,000*3,432	6,000*0,400	4,000*0,432	0,000*2,747	0,000*0,000	-1,000*0,000	0,000*0,000	0,000*0,000	0,000*13,989	0,000*0,000	0,000*0,000	<=	13,000	(12,105)
2,000*0,084	11,000*0,137	0,000*3,432	4,000*0,400	4,000*0,432	0,000*2,747	0,000*0,000	0,000*0,000	-1,000*0,000	0,000*0,000	0,000*13,989	0,000*0,000	0,000*0,000	<=	5,000	(5,000)
3,000*0,084	2,000*0,137	1,000*3,432	5,000*0,400	3,000*0,432	0,000*2,747	0,000*0,000	0,000*0,000	0,000*0,000	-1,000*0,000	0,000*13,989	0,000*0,000	0,000*0,000	<=	8,000	(7,253)
2,000*0,084	3,000*0,137	9,000*3,432	7,000*0,400	4,000*0,432	0,000*2,747	0,000*0,000	0,000*0,000	0,000*0,000	0,000*0,000	-1,000*13,989	0,000*0,000	0,000*0,000	<=	22,000	(22,000)
2,000*0,084	3,000*0,137	2,000*3,432	3,000*0,400	4,000*0,432	0,000*2,747	0,000*0,000	0,000*0,000	0,000*0,000	0,000*0,000	0,000*13,989	-1,000*0,000	0,000*0,000	<=	14,500	(10,368)
7,000*0,084	5,000*0,137	1,000*3,432	5,000*0,400	3,000*0,432	0,000*2,747	0,000*0,000	0,000*0,000	0,000*0,000	0,000*0,000	0,000*13,989	0,000*0,000	-1,000*0,000	<=	8,000	(8,000)
2,000*0,084	1,000*0,137	1,000*3,432	1,000*0,400	2,000*0,432	0,000*2,747	0,000*0,000	0,000*0,000	0,000*0,000	0,000*0,000	0,000*13,989	0,000*0,000	0,000*0,000	>=	5,000	(5,000)
1,000*0,084	0,000*0,137	2,000*3,432	2,000*0,400	1,000*0,432	0,000*2,747	0,000*0,000	0,000*0,000	0,000*0,000	0,000*0,000	0,000*13,989	0,000*0,000	0,000*0,000	<=	15,000	(8,179)
0,000*0,084	1,000*0,137	1,000*3,432	5,000*0,400	1,000*0,432	0,000*2,747	0,000*0,000	0,000*0,000	0,000*0,000	0,000*0,000	0,000*13,989	0,000*0,000	0,000*0,000	>=	6,000	(6,000)
2,000*0,084	2,000*0,137	3,000*3,432	1,000*0,400	2,000*0,432	0,000*2,747	0,000*0,000	0,000*0,000	0,000*0,000	0,000*0,000	0,000*13,989	0,000*0,000	0,000*0,000	>=	12,000	(12,000)
4,000*0,084	1,000*0,137	1,000*3,432	0,000*0,400	1,000*0,432	0,000*2,747	0,000*0,000	0,000*0,000	0,000*0,000	0,000*0,000	0,000*13,989	0,000*0,000	0,000*0,000	<=	24,000	(4,337)

Рисунок 5.3 - Приклад виводу системи: ілюстрація обмежень, пов’язаних із суттєвими та несуттєвими частковими функціоналами.

Для підтвердження цього було прийняте рішення про проведення експериментів, у яких зміни будуть вноситись у обмеження пов’язані із суттєвими та несуттєвими частковими функціоналами окремо, а потім, за допомогою зібраних статистичних даних буде зроблено висновок, наскільки великий вплив мають суттєві та несуттєві обмеження на результати розв’язання задачі.

Результати першого типу експериментів

У результаті проведення першого типу експериментів з’явилась гіпотеза про існування суттєвих та несуттєвих часткових функціоналів та пов’язаних із ними обмежень у задачах лінійного програмування в умовах невизначеності.

5.4. Другий тип експериментів

Другий тип експериментів спрямований на отримання результатів, які б підтвердили існування суттєвих обмежень у задачах лінійного програмування в умовах невизначеності.

Для цього, на відміну від попередніх експериментів, зміни будуть вноситись виключно цілеспрямовано – лиш у обмеження, пов’язані із суттєвими частковими функціоналами. Нагадаємо, суттєвими функціоналами вважаються ті, у яких значення суми відповідного обмеження достатньо близька до числового значення обмеження, у ході експериментів близькими вважаються ті вектори, значення суми яких знаходиться у межах 1%-ї похибки відносно числового значення обмеження.

Для перевірки гіпотези використано генератор, який згенерував 50 задач, а потім у суттєві обмеження цих задач вносилась похибка із значенням 1,2,3,…10%. Якщо розв'язок або значення цільової функції у задачі відрізнялось від початкового, то це фіксувалось у пам'яті програми та виводилось у кінцевих звітах та на екранній формі головного меню.

У результаті 50 генерацій було сформовано статистику і створено файл звіту у форматі .csv. Для зручності, дані було відображено у табличному вигляді.

На рисунку 5.4 видно, що для кожного із варіантів зміни: при збільшенні чи зменшені значень суттєвих обмежень, значення вектору розв'язку змінилося у 41 із 50 випадків.

На графіку рисунку 5.4 синій стовпчик відповідає за ті випадки, коли зміна була від'ємна: тобто значення обмеження зменшувались на певний відсоток, а помаранчевий – коли зміна була додатна: коли значення збільшувались на певний відсоток. Номер стовпчика відповідає відсотку зміни значення.

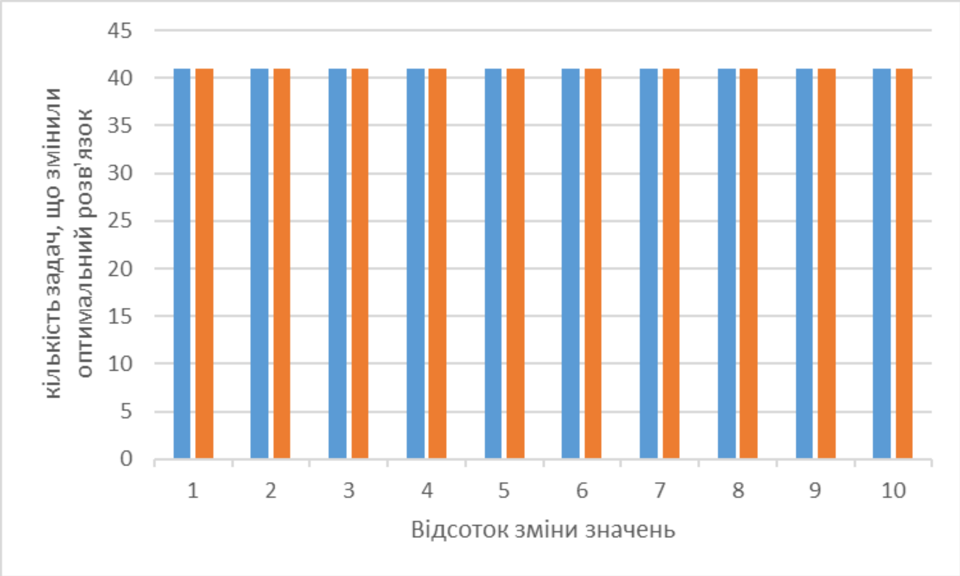


Рисунок 5.4 – Статистика результатів проведення 2-го типу експериментів, генерація 50-ти задач. Тут синій стовпчик відповідає за ті випадки, коли зміна значень була від'ємна: тобто значення обмеження зменшувались на певний відсоток, а помаранчевий – коли зміна була додатна: коли значення збільшувались на певний відсоток. Номер стовпчика відповідає відсотку зміни значення.

Оскільки, приблизно у 20% випадків зміни не призвели до зміни розв'язку, можна висунути гіпотезу, що зміна значень векторів не завжди змінює розв'язок. Причиною цьому можуть слугувати достатньо жорсткі обмеження часткових задач, або достатньо широкі межі, задані експертами для задачі в умовах невизначеності.

Для перевірки результатів було виконано ще один запуск програми, результати приведено на рисунку 5.5.

Згідно із результатами цього прогону, при достатньо великому відхиленні значень від початкових, усі задачі мають шанс змінити значення вектору розв'язку X.

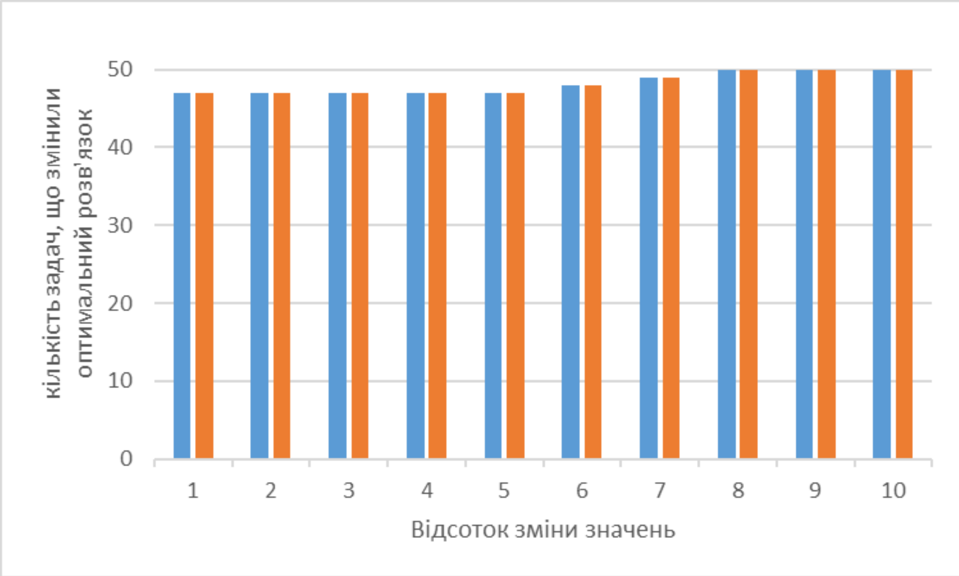


Рисунок 5.5 - Статистика результатів проведення 2-го типу експериментів, генерація 50-ти задач, друга спроба. Тут синій стовпчик відповідає за ті випадки, коли зміна значень була від'ємна: тобто значення обмеження зменшувались на певний відсоток, а помаранчевий – коли зміна була додатна: коли значення збільшувались на певний відсоток. Номер стовпчика відповідає відсотку зміни значення.

Із отриманих у ході інших прогонів результатів було вирахувано шанс того, що зміна обмеження пов'язаного із суттєвим функціоналом вплине на результати обчислень, ці дані приведено нижче у таблиці 5.1.

Таблиця 5.1 – Узагальнена статистика зміни розв'язку задачі в умовах невизначеності, залежно від відсотку зміни обмежень, пов'язаних із суттєвим функціоналом.

Відсоток зміни значень суттєвих обмежень	Шанс зміни, у %	
	При збільшені значень суттєвого обмеження	При зменшені значень суттєвого обмеження
1	77.2	77.2
2	77.6	77.6
3	77.6	77.6
4	78.0	78.0
5	78.0	78.0
6	78.8	78.8

Продовження таблиці 5.1

7	79.2	79.2
8	81.6	81.6
9	81.6	81.6
10	81.6	81.6

У середньому, зміна значень суттєвих обмежень призводить до зміни значень розв'язку або/та значень цільової функції у 79.12% випадків. Це достатньо велике значення, через що можна зробити висновок, що обмеження із сумарними значеннями близькими до числового значення обмеження і справді пов'язані із суттєвими частковими функціоналами, тобто мають сильний вплив на результати розв'язання, навіть при незначних змінах.

Було помічено, що для задач, які були згенеровані експертами, статистика приймає інший вигляд. Причиною цьому може слугувати те, що у генератора задач задіяно рівномірний розподіл, коли на практиці оцінки експертів одної задачі в умовах невизначеності мають несуттєві відмінності. Так, якщо певний параметр має суттєвий вплив, то в усіх експертів відповідне йому значення із вектора коефіцієнтів буде великим. Не обов'язково усі ці оцінки матимуть однакове значення, проте значення буде виділятись серед інших коефіцієнтів у векторах часткових функціоналів.

Генератор же має рівномірний розподіл. При рівномірному розподілі значення може бути лиш у певному діапазоні, отже, не обов'язково оцінки будуть еквівалентні. Звісно, це відповідає невизначеності у широкому сенсі, проте випадкові параметри не зможуть слугувати достовірною оцінкою задачі.

Так, для другого типу експериментів при генерації часткових функціоналів експертом статистика приведена на рисунку 5.6:

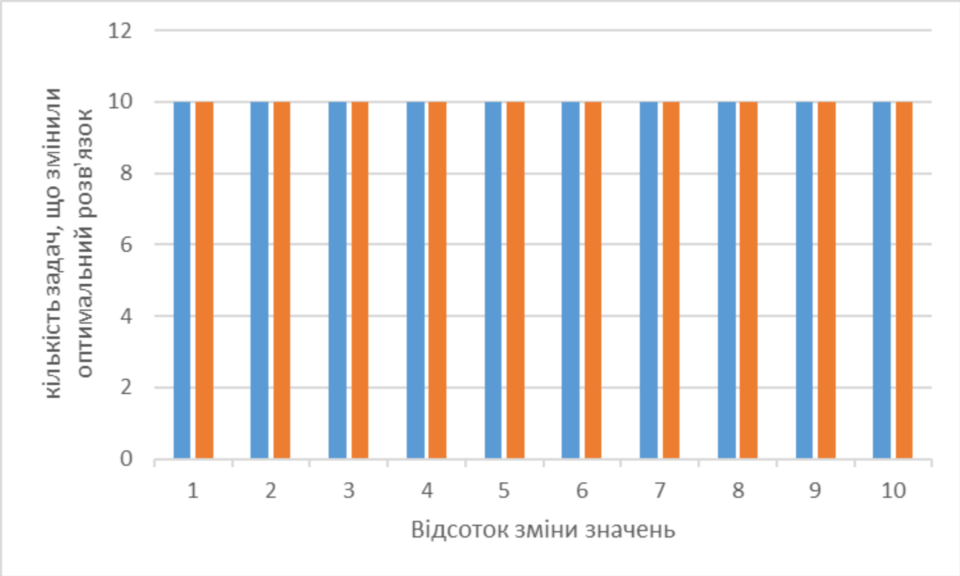


Рисунок 5.6 Статистика зміни задачею розв’язку при зміні суттєвих обмежень, генерація експертом 10-ти задач. Тут синій стовпчик відповідає за ті випадки, коли зміна була від’ємна: тобто значення обмеження зменшувались на певний відсоток, а помаранчевий – коли зміна була додатна: коли значення збільшувались на певний відсоток. Номер стовпчика відповідає відсотку зміни значення.

Як видно по гістограмі на рисунку 5.6 – зміна суттєвих векторів призводила до зміни розв’язку задачі у 10 випадках із 10, якщо задача була згенеровані експертом.

Задачею у даному випадку є повноцінна задача лінійного програмування в умовах невизначеності. Усі вектори коефіцієнтів були побудовані за оцінками експертів і кожна із задач була унікальною.

Таблична презентація результатів викладена у Таблиці 5.2.

Таблиця 5.2 – Статистика зміни розв’язку задачі в умовах невизначеності, залежно від відсотку зміни суттєвих обмежень при генерації задач людиною.

Відсоток зміни значень несуттєвих обмежень	Шанс зміни, у %	
	При збільшені значень несуттєвого обмеження	При зменшені значень несуттєвого обмеження
1	100	100
2	100	100
3	100	100
4	100	100
5	100	100
6	100	100
7	100	100
8	100	100
9	100	100
10	100	100

У середньому, зміна значень суттєвих обмежень призводить до зміни значень розв’язку або/та значень цільової функції у 100% випадків. Це достатньо велике значення, через що можна зробити висновок, що обмеження, сумарне значення яких було достатньо близьким до чисельного значення обмеження – і справді пов’язані із суттєвими частковими функціоналами, тобто мають сильний вплив на результати розв’язання, навіть при незначних змінах.

Результати другого типу експериментів

У результаті проведення експериментів було встановлено, що у задачах лінійного програмування в умовах невизначеності існують обмеження, пов’язані із суттєвими частковими функціоналами, зміна яких із достатньо високою ймовірністю веде до зміни розв’язку задачі.

Було помічено, що між задачами, згенерованими людиною і генератором на основі рівномірного закону розподілу існує суттєва різниця у статистиці. Це можна пояснити тим, що оцінки експертів, які створюються людиною мають певну міру еквівалентності. При цьому, оцінки, створені генератором – випадкові.

5.5. Третій тип експериментів

Третій тип експериментів спрямований на доведення того, що не усі обмеження мають істотний вплив на результати розв'язання, тобто не усі обмеження пов'язані із суттєвими частковими функціоналами.

Для перевірки цього твердження, у 3-му типі експериментів випадкові зміни вносяться виключно до обмежень, пов'язаних із несуттєвими частковими функціоналами.

У результаті першого прогону було отримано результати, зображені на рисунку 5.7.

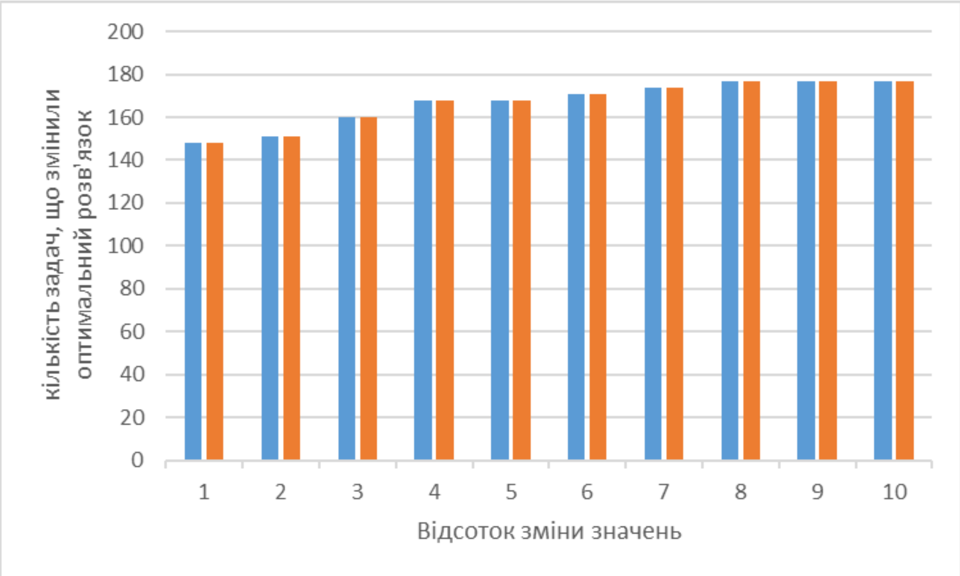


Рисунок 5.7 – Статистика результатів проведення 3-го типу експериментів, генерація 200-ти задач. Тут синій стовпчик відповідає за ті випадки, коли зміна була від'ємна: тобто значення обмеження зменшувались на певний відсоток, а помаранчевий – коли зміна була додатна: коли значення збільшувались на певний відсоток. Номер стовпчика відповідає відсотку зміни значення.

На рисунку 5.7 зображено кількість задач, які змінили значення розв'язку при внесенні змін у значення обмежень, пов'язаних із несуттєвими частковими функціоналами. Для достовірності отриманих результатів було виконано ще декілька прогонів, та було зібрано статистичні дані, зображені на рисунку 5.8.

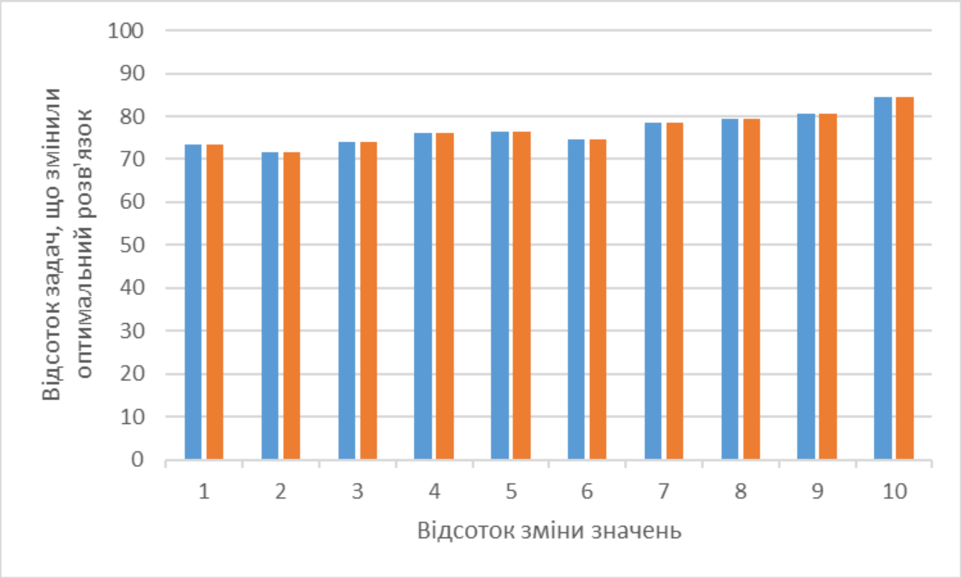


Рисунок 5.8 – Середнє відсоткове значення шансу зміни задачею розв’язку, при зміні несуттєвих обмежень. Тут синій стовпчик відповідає за ті випадки, коли зміна була від’ємна: тобто значення обмеження зменшувались на певний відсоток, а помаранчевий – коли зміна була додатна: коли значення збільшувались на певний відсоток. Номер стовпчика відповідає відсотку зміни значення. Числові значення подано в таблиці 5.3.

Таблиця 5.3 – Узагальнена статистика зміни розв’язку задачі в умовах невизначеності, залежно від відсотку зміни несуттєвих обмежень.

Відсоток зміни значень несуттєвих обмежень	Шанс зміни, у %	
	При збільшені значень несуттєвого обмеження	При зменшені значень несуттєвого обмеження
1	73,5	73,5
2	71,5	71,5
3	74	74
4	76	76
5	76,5	76,5

Продовження таблиці 5.3.

6	74,5	74,5
7	78,5	78,5
8	79,5	79,5
9	80,5	80,5
10	84,5	84,5

У середньому, зміна значень несуттєвих обмежень призводить до зміни значень розв’язку або/та значень цільової функції у 76.9% випадків. Це досить велике значення. Через це можна було б зробити висновок, що насправді не існує ніяких суттєвих чи несуттєвих часткових функціоналів. Проте із попереднього експерименту було зроблено висновок, що генератор не може слугувати джерелом достовірних даних. Для порівняння отриманих результатів було виконано збір статистичних даних при генерації задач експертами.

Як і у попередній серії експериментів, було помічено, що оцінки, які було створено генератором мають суттєву відмінність у статистиці, порівняно зі створеними експертами.

Для задач, згенерованих експертом, а не генератором із рівномірним розподілом, статистика представлена на рисунку 5.9.

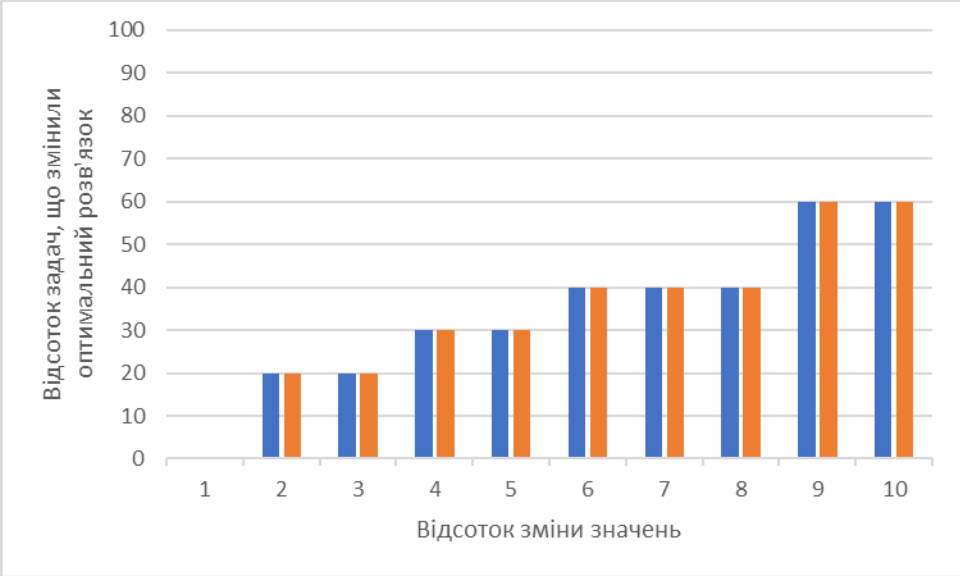


Рисунок 5.9 – Середнє відсоткове значення шансу зміни задачею розв’язку, при зміні несуттєвих обмежень. Числові значення подано в таблиці 5.4.

Таблиця 5.4 – Узагальнена статистика зміни розв’язку задачі в умовах невизначеності, залежно від відсотку зміни несуттєвих обмежень.

Відсоток зміни значень несуттєвих обмежень	Шанс зміни, у %	
	При збільшені значень несуттєвого обмеження	При зменшені значень несуттєвого обмеження
1	0	0
2	20	20
3	20	20
4	30	30
5	30	30
6	40	40
7	40	40
8	40	40
9	60	60
10	60	60

У середньому, зміна значень суттєвих обмежень призводить до зміни значень розв’язку або/та значень цільової функції у 34% випадків. Як і було передбачено, отримані у результаті генерації оцінки недостовірні, через що якість отриманих даних суттєво відрізняється. Так, різниця між суттєвими та несуттєвими частковими функціоналами стає помітною лиш на даних, згенерованих експертом. Із отриманих статистичних даних можна зробити висновок, що у задачах лінійного програмування в умовах невизначеності і справді існують несуттєві часткові функціонали, і зміна обмежень, пов’язаних із ними має меншу міру впливу на результати розв’язання, аніж у суттєвих. Нарешті, було підтверджено, що між суттєвими і несуттєвими частковими функціоналами існує різниця, що доводить гіпотезу про існування обох видів часткових функціоналів.

Результати третього типу експериментів

У результаті проведення експериментів третього типу було підтверджено, що найбільший вплив на результати мають зміну у обмеженнях суттєвих часткових функціоналів, коли внесення змін у обмеження несуттєвих, здебільшого не несе впливу на результати розв’язання.

Також було підтверджено, що між задачами, згенерованими експертами та задачами, згенерованими генератором існує істотна різниця, яка не дозволяє використовувати генератор для отримання достовірних оцінок і задач.

Висновки до розділу

У розділі дано роз’яснення про 3 типи експериментів, які було проведено у ході магістерського дослідження. Показано суттєву різницю у статистиці, зібраній для даних, згенерованих експертом та генератором, котра вказала на неможливість використання генератора як джерела якісних даних.

Результатом проведення першого типу експериментів стала поява гіпотези про існування суттєвих та несуттєвих часткових функціоналів, про зв’язок між цими функціоналами та відповідними їм обмеженнями. Для підтвердження цієї гіпотези складено два типи експериментів: другого і третього типу.

Результатом виконання другого типу експериментів стало підтвердження частини гіпотези про існування обмежень, пов’язаних із суттєвими частковими функціоналами. Для винесення кінцевого висновку третій тип експериментів було скореговано.

У ході третього типу експериментів було підтверджено, що існують обмеження, пов’язані із несуттєвими частковими функціоналами. Отриманий результат надав можливість підтвердити істинність гіпотези.

Також, дослідження показало, що генерація даних для задач лінійного програмування в умовах невизначеності має бути виконана виключно з допомогою експертів, оскільки результати, отримані при використанні задач, отриманих при автоматичній генерації – були недостовірні.

6 СТАРТАП ПРОЕКТ

Назва проекту

Розроблено стартап-проект на тему «**Автоматизовані збиральні лінії»**.

6.1 Короткий опис проекту

Проект дозволяє підприємцям швидко розгортати виробничі лінії легкої промисловості для виробництва власної продукції.

6.2 Бізнес-модель

6.2.1 Цінний продукт

Цінними якостями проекту «Автоматизовані збиральні лінії» відмінними від існуючих є (фактори створення цінності):

- новизна: проект не має близьких аналогів або аналоги не відомі на ринку;
- дозволяє не орієнтувати бізнес відносно існуючої продукції, а створювати продукцію, залежно від вимог бізнесу;
- при масовому виробництві дозволяє зменшити витрати, порівняно із закупівлею готової продукції від виробника;

6.2.2 Сегмент споживачів

Проект націлений на малий та середній сегмент бізнесу, орієнтованого на виробництво та продаж, оскільки цей проект дозволить автоматизувати виробництво та зменшити його вартість.

6.2.3 Канали збуту

Каналами збуту є:

- виставки для малого і середнього бізнесу, де серед інших новинок можна презентувати проект «Автоматизовані збиральні лінії»;
- продажі з офіційного сайту стартап-проекту.

6.2.4 Взаємодія з споживачами

Залучення нових клієнтів стимулюється клієнтською програмою через запрошення нових клієнтів старими із отриманням останніми бонусів чи знижок;

Підтримка користувачів здійснюється через:

- інтерактивний сайт «Автоматизовані збиральні лінії»;
- телефон;
- мессенджери;
- електронну пошту.

6.2.5 Дохід (монетизація)

Дохід буде отримуватись шляхом прямих продажів обладнання або шляхом передачі обладнання у ренту.

6.2.6 Ключові види діяльності

Ключовим видом діяльності стартапу буде здійснення виробництва та покращення обладнання «Автоматизовані збиральні лінії». Для цього буде виконано:

- *розробницька діяльність* - *з*акінчення розробки сайту, креслень, концептів;
- *наукова діяльність* - проведення досліджень, націлених на покращення якості і універсальності обладнання;
- *маркетингова діяльність* - збільшення каналів донесення інформації до потенційних користувачів.

6.2.7 Ключові ресурси

Ключовими ресурсами системи є :

- *програмні* (виготовлені кваліфікованими робітниками);
- *матеріальні* (виробниче обладнання; обладнання, яке виробляється для продажу; складські приміщення; виробничі приміщення);
- *інтелектуальні ресурси* (будуть використані власні креслення, патенти, технічні розробки).

6.2.8 Людські ресурси

Людськими ресурсами даної системи є

- *директор* – маркетолог із вищою освітою і досвідом роботи не менше 4 роки, менеджер управлінець;
- *менеджер* з продаж з досвідом не менше 5 років;
- *бухгалтер;*
- *юрист з досвідом не менше 5 років;*
- *маркетолог* з досвідом роботи не менше 2 роки;
- *IT розробники*;
- *Інженери*;
- *персонал складу*.

6.2.9 Витрати

Виготовлення та реалізація даної системи:

- повна собівартість реалізованої продукції стартапу – 250000 грн;
- прибуток стартапу – 1000000 грн;
- оптова ціна виробника – 30000 грн;
- непрямі податки: - Податок на додану вартість – 10000 грн;
- оптова ціна відпускна – 30000 грн.

6.3 Споживчі властивості товару

- професійне спеціалізоване програмне забезпечення для управління системами виробництва;
- ПЗ дає змогу управляти даною системою дистанційно.

6.4 Дослідження ринку

Ідея розробки стартапу, який дозволяє розгортати власне виробництво – нова, проте потреба у такому стартапі висока. Причиною цьому слугує низька зацікавленість виробників у кастомізації продукції, що змушує бізнес або повністю орієнтуватись на виробника, або створювати власні системи та методи виробництва, які мають високу ціну та малу ефективність.

6.5 Дослідження конкурентного оточення

Конкурентне оточення продажу аналогів стартапу «Автоматизовані збиральні лінії»: виробничі потужності Китаю.

6.6 Маркетингова стратегія просування

Новий товар на існуючий ринок.

6.7 Елементи фінансового плану

6.7.1 Опис бізнес-проекту

Організація виробництва та реалізації у формі стартап-виробничої фірми у відповідності до заявленого проекту.

6.7.2 Опис товару/послуги/технології

Стартап призначений надати змогу підприємцям швидко розгортати виробничі лінії легкої промисловості для виробництва власної продукції.

6.7.3 Маркетинг та продаж

Дослідження ринку показало, що даний проект є першим на ринку. Поки що у підприємств немає можливості придбати універсальне обладнання для усіх необхідних видів робіт по виробництву за низьку ціну. Єдиними способами є або придбання вузькоспеціалізованого та коштовного обладнання, або виконання виробництва засобами людської праці.

6.7.4 Фінансовий план

Витрати на організацію стартапу на період 9 місяців:

- організацію виробничої ділянки, електропостачання, виробничого приміщення, закупівля обладнання, організація складу матеріалів та інструментів, умов безпечної роботи, охорони - 40 000 грн;
- створення робочої команди стартапу, прийом на роботу робітників – 40000 грн;

- закупівля матеріалів та інструментів – 30000 грн;
- розробка робочих креслень та креслень збірки продукту – 50000 грн;
- оренда приміщення – 15000 грн;
- комунальні платежі – 15000 грн;
- резерв, непередбачувані витрати - 10000 грн.

6.7.5 Резюме

Продукцію стартапу планується поширювати за рахунок прямих продажів та через залучення нових користувачів попередніми клієнтами. Менеджер з продажу демонструватиме керівнику підприємства можливості обладнання, яке підприємство матиме змогу орендувати або придбати. Передбачається розробка спеціальних замовлень для постійних клієнтів відповідно до їх потреб.

6.8 Презентація проекту інвестору

6.8.1 Ідея проекту

Ідея – створення уніфікованого обладнання для швидкого розгортання виробництва. Продаж або здача у ренту цього обладнання малому та середньому бізнесу, орієнтованому на виробництво та продаж.

6.8.2 Опис проблеми або можливості

На світовому ринку відсутні аналоги із заявленими можливостями та властивостями. Через це висока ймовірність того, що ідея перший час матиме низький успіх.

З часом, коли недовіра до стартапу спаде, прогнозується швидкий ріст зацікавленості у ідеї.

6.8.3 Рішення

В результаті використання продукту стартапу «Автоматизовані збиральні лінії» за призначенням споживач має можливість:

− швидко розгорнути виробничу діяльність за власними потребами;

- швидко і зручно змінити конфігурацію обладнання;
- наростити або скоротити об’єм виробництва;

6.8.4 Конкуренти

Конкурентне оточення продажу аналогів стартапу «Автоматизовані збиральні лінії» –виробничі потужності Китаю.

6.8.5 Ринок

Новий товар на старий ринок.

6.8.6 Маркетингова стратегія

Активне просування через сайт.

Активне просування через участь у презентаціях та конференціях.

Активне просування через прямі продажі.

6.8.7 Поточна ситуація

В даний час розробляється сайт - основна платформа для реєстрації користувачів та ідеї для створення обладнання.

6.8.8 Команда проекту

Попередні домовленості про участь 2-х потенційних робітників.

6.8.9 Фінансові показники

Попередньо, на найближчі 5 місяців витрати відсутні. Ведеться розробка концептів та ідей.

6.8.10 Пропозиція інвестору

Вкласти угоду на отримання 10% від продажу або сдачі у ренту обладнання, у створення якого було покладено інвестиції. Також планується проведення IPO та емісія акцій, 49% яких буде передано для закупівлі інвесторами.

6.9 Подальші кроки в проекті

6.9.1 Наукова діяльність

Проведення досліджень для покращення точності обладнання та покращення ергономічності програмних застосунків, які дозволяють працювати із обладнанням навіть користувачам без необхідних знань у галузі виробництва.

6.9.2 Розробницька діяльність

Закінчення розробки сайту, концепту обладнання та програмного забезпечення для нього.

Висновки до розділу

В цьому розділі було подано опис стартап-проекту «Автоматизовані збиральні лінії». Цей стартап - новий в своєму роді, що дає непогані шанси на успіх за рахунок відсутності конкурентів, проте не виключає ризику відторгення суспільством через новизну ідеї. Кожен підприємець, у тій чи іншій мірі, бажає створення свого особливого і неповторного продукту, проте обмежений у засобах досягнення цього бажання. Описаний у розділі стартап дозволить реалізувати це бажання на практиці.

ВИСНОВОК ДО РОБОТИ

У результаті роботи над цим магістерським дослідженням, було розроблено систему, яка дозволила виконувати дослідження властивостей задач лінійного програмування в умовах невизначеності.

У результаті виконання досліджень властивостей задач лінійного програмування в умовах невизначеності було встановлено, що у цього класу задач існують суттєві та несуттєві часткові функціонали, які мають наступні властивості:

– зміна обмежень додаткової задачі лінійного програмування, пов’язаних із суттєвими частковими функціоналами, приводить до зміни оптимального рішення додаткової задачі лінійного програмування;

– зміна обмежень додаткової задачі лінійного програмування, пов’язаних із несуттєвими частковими функціоналами, або не приводить до зміни оптимального рішення додаткової задачі лінійного програмування, або призводить, лиш коли ця зміна перевищує певний поріг, індивідуальний для кожної окремої задачі.

Згідно із отриманими результатами, зміна обмежень, пов’язаних із суттєвими частковими функціоналами вже при зміні цих обмежень на 1% відносно початкових призводить до зміни задачею оптимального розв’язку у 100% випадків, якщо задачі згенеровані експертом. Для задач, отриманих при роботі генератора, при 1% змін цей показник складає 77.2% і поступово зростає до 81.6% при зміні у 10% від початкового значення обмеження.

При цьому, за статистикою, зміна обмежень, пов’язаних із несуттєвими частковими функціоналами починає змінювати оптимальний розв’язок задач, згенерованих експертом лиш коли зміни перетинають поріг у 2% і лиш у 20% задач. Цей показник поступово зростає, і при розмірі змін у 10% лиш 60% задач змінюють свій оптимальний розв’язок. Для задач, отриманих у результаті роботи генератора результати статистики інші. Для таких задач при 1% змін у середньому 73.5% задач змінюють оптимальний розв’язок. При цьому відсоток задач, які змінюють свій розв’язок при внесенні змін у обмеження, пов’язані із несуттєвими частковими функціоналами, не зростає поступово, а має непередбачувані коливання: при розмірі змін у 2% кількість задач, що змінила свій оптимальний розв’язок різко знижується до 71.5%, зростає до 76% на 4% змін і знову спадає до 74.5% на 5%, у подальшому зростаючи до 84.5% на 10%. Для дослідження цього явища було виконано збір статистики для 200 задач, що виключає можливість суттєвого впливу від деякої кількості задач на результат. Така статистика можлива лиш коли кількість задач, які мають такі властивості достатньо велика.

Із описаного вище було встановлено, що при генерації значень часткових функціоналів задач, з використанням у генераторі рівномірного розподілу, отримані значення не можуть у повній мірі імітувати оцінки експертів. Це пов’язано із тим, що оцінки різних експертів однієї задачі – еквівалентні: тобто числові характеристики часткових функціоналів можуть відрізнятись, але ця різниця невелика, або співпадає у більшості експертів.

ПЕРЕЛІК ПОСИЛАНЬ

1. Pavlov A.A. Optimization for one class of combinatorial problems under uncertainty. Адаптивні системи автоматичного управління. 2019. 1. № 34. С. 81–89. doi: 10.20535/1560-8956.1.2019.178233.

2. Pavlov A.A. Combinatorial optimization under uncertainty and formal models of expert estimation. Вісник Національного технічного університету «ХПІ». 2019. № 1. С. 3–7. doi: 10.20998/2079-0023.2019.01.01.

3. Alexander A. Pavlov, Elena G. Zhdanova The Transportation Problem under Uncertainty // Journal of Automation and Information Sciences – 2020. – № 4 (52)– pp. 1-13. (Scopus)

4. Pavlov A. Zhdanova E. Finding a compromise solution to the transportation problem under uncertainty // Adaptive systems of automatic control. . – 2020. – № 1 (36)– С.60-72. (Фахове видання)

5. Yurii Selin, Tatyana Shulkevych, Mykola Bohdanenko and Olena Zhdanova. Information Technology of Data Mining and Forecasting of Nonlinear Non-Stationary Processes. 2020 IEEE 2nd International Conference on System Analysis & Intelligent Computing (SAIC), 05-09 October, 2020 Kyiv, Ukraine, pp. 309-315, IEEE Catalog Number: CFP20SUA-CDR ISBN: 978-1-7281-9083-9/20 ©2020 IEEE

6. Богданенко М.О. "С*истема підтримки досліджень оптимізаційних задач в умовах невизначеності*" / О.Г. Жданова, О.А. Павлов // Матеріали V всеукраїнської науково-практичної конференції молодих вчених та студентів "Інформаційні системи та технології управління" (ІСТУ-2020) - м. Київ.: НТУУ "КПІ ім.Ігоря Сікорського", 26-27 листопада 2020 р.

7. Богданенко М.О., Селін Ю.М., Автоматизація процесу підтримки прийняття рішень з формування стратегій видобування криптовалюти

8. Zgurovsky M.Z., Pavlov A.A. Combinatorial Optimization Problems in Planning and Decision Making: Theory and Applications, 1st edn. Studies in

Systems. *Decision and Control*. 173, Springer, Cham. 2019. 526 pp. doi: 10.1007/978-3-319-98977-8.

9. Згуровский М.З., Павлов А.А. Труднорешаемые задачи комбинаторной оптимизации в планировании и принятии решений. Киев : Наук. думка. 2016. 716 с.

10. Ногин В.Д. Линейная свертка критериев в многокритериальной оптимизации. *Искусственный интеллект и принятие решений.* 2014. № 4. С. 73–82.

11. Gorelik V.A., Zolotova T.V. Problem of selecting an optimal portfolio with a probabilistic risk function. 2016.. –. https://doi.org/10.1007/s10958-016-2921-z.

12. Kharchenko A., Halay I., Zagorodna N., Bodnarchuk I. Trade-off optimal decision of the problem of software system architecture choice. –. http://doi.org/10.1109/STC-CSIT.2015.7325465.

13. Новикова Н.М., Поспелова И.И., Зенюков А.И. Метод сверток в многокритериальных задачах с неопределенностью–. http://doi.org/10.7868/S0002338817050031.

14. Bindima T., Elias E. A novel design and implementation technique for low complexity variable digital filters using multi-objective artificial bee colony optimization and a minimal spanning tree approach. 2017.. –. https://doi.org/10.1016/j.engappai.2016.12.011.

15. Mirzapour S.M.J., Hashem Al.-E., Malekly H., Aryanezhad M.B. A multi-objective robust optimization model for multi-product multi-site aggregate production planning in a supply chain under uncertainty. 2011. . P.. https://doi:10.1016/j.ijpe.2011.01.027.

16. Ємець О.О., Барболіна Т.М. Лінійні оптимізаційні задачі на розміщеннях з імовірнісною невизначеністю: властивості і розв'язання 2016. . https://doi.org/10.20535/SRIT.2308-8893.2016.1.11.

17. Таха Х.А. Введение в исследование операций. М. : Вильямс, 2005. 912 с.

18. Таха, Хемди А., Введение в исследование операций, 7-е издание.: Пер. с англ. — Москва: Издательский дом "Вильяме", 2005. — 912 с. ISBN 5-8459-0740-3

19. Катренко А.В. Теорія прийняття рішень [Текст] / А.В. Катренко, В.А. Пасічник, В.П. Пасько – Л. : Новий світ – 2000, 2009. – 396 с.

20. Кини Р. Принятие решений при многих критериях, предпочтениях и замещениях: пер. с англ. / Р. Кини, Х. Райфа. – М. : Радио и связь, 1981. – 560 с. 266

21. Киселева Е.М. Непрерывные задачи оптимального разбиения множеств: теория, алгоритмы, приложения: монография / Е.М. Киселева, Н.З. Шор. – К. : Наук. думка, 2005. – 564 с.

22. Кисельова О.М., Методичні вказівки до виконання лабораторних робіт з курсу «Математичне моделювання економічних процесів». / О.М. Кисельова, Л.С. Коряшкіна; М-во освіти і науки України, Дніпропетр. нац. ун-т. – Д. : ДНУ, 2008. – 48 с.

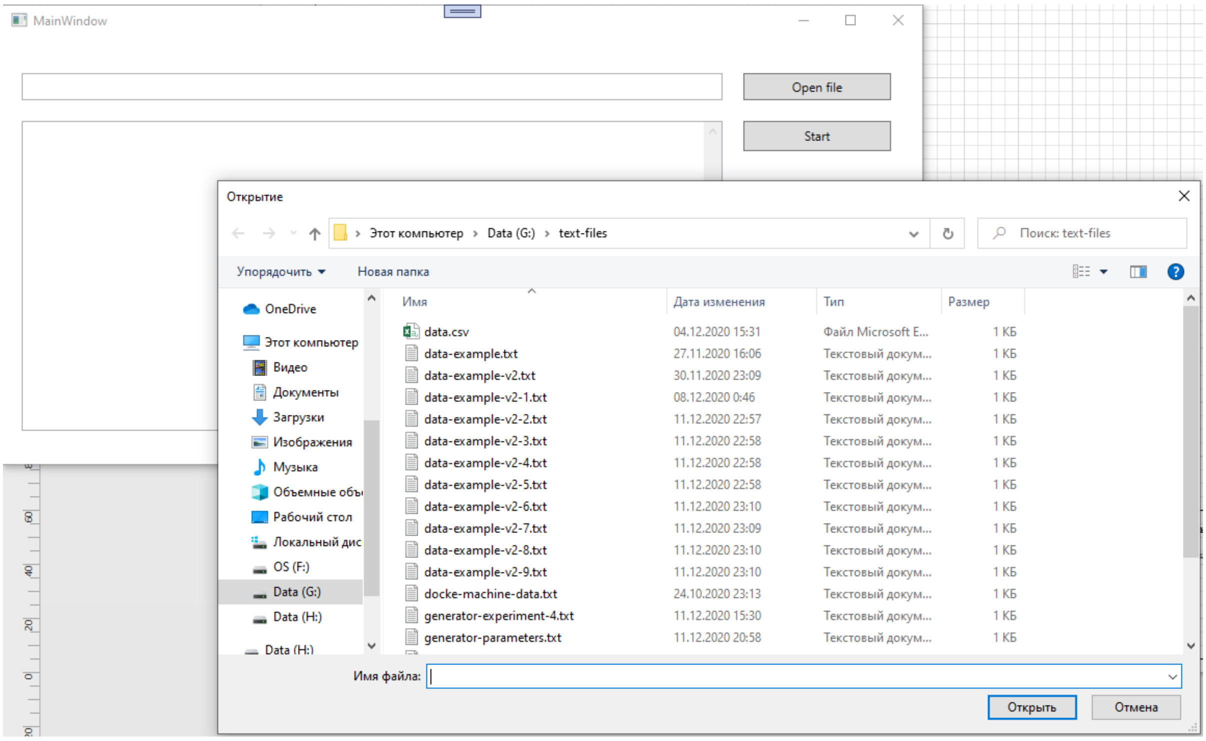
23. Волошин О.Ф., . «Моделі та методи прийняття рішення». / С.О. Мащенко; – 2-ге вид., перероб. та допов. – К. : Видавничо-поліграфічний центр "Київський університет", 2010. – 336 с.

ДОДАТОК А

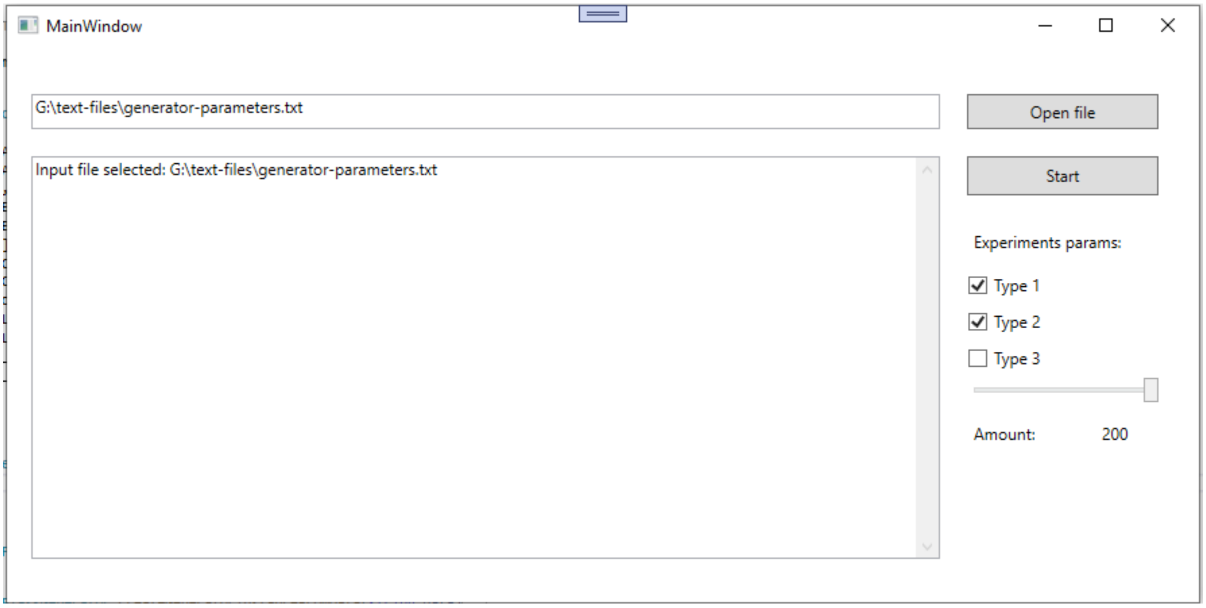
Графічний матеріал

Плакат 1 - Екранна форма головного меню системи до вибору файлу даних

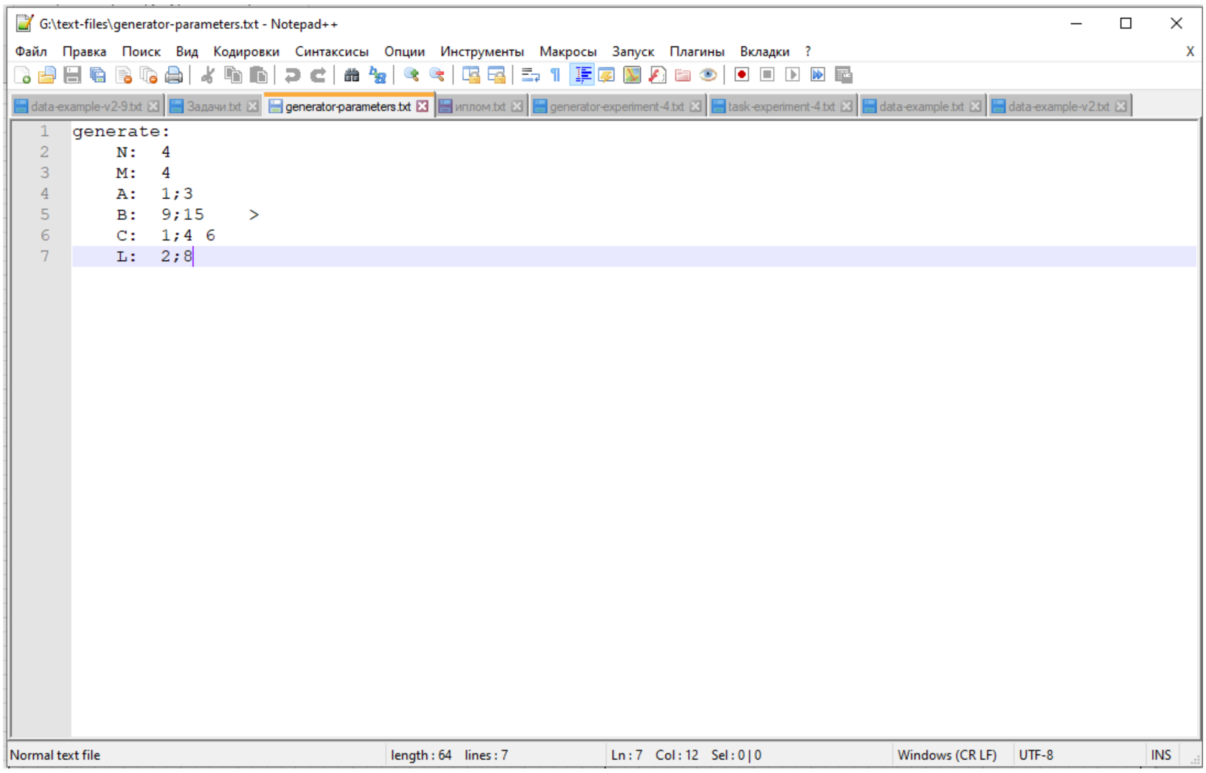
Плакат 2 - Екранна форма меню вибору файлу вхідних даних



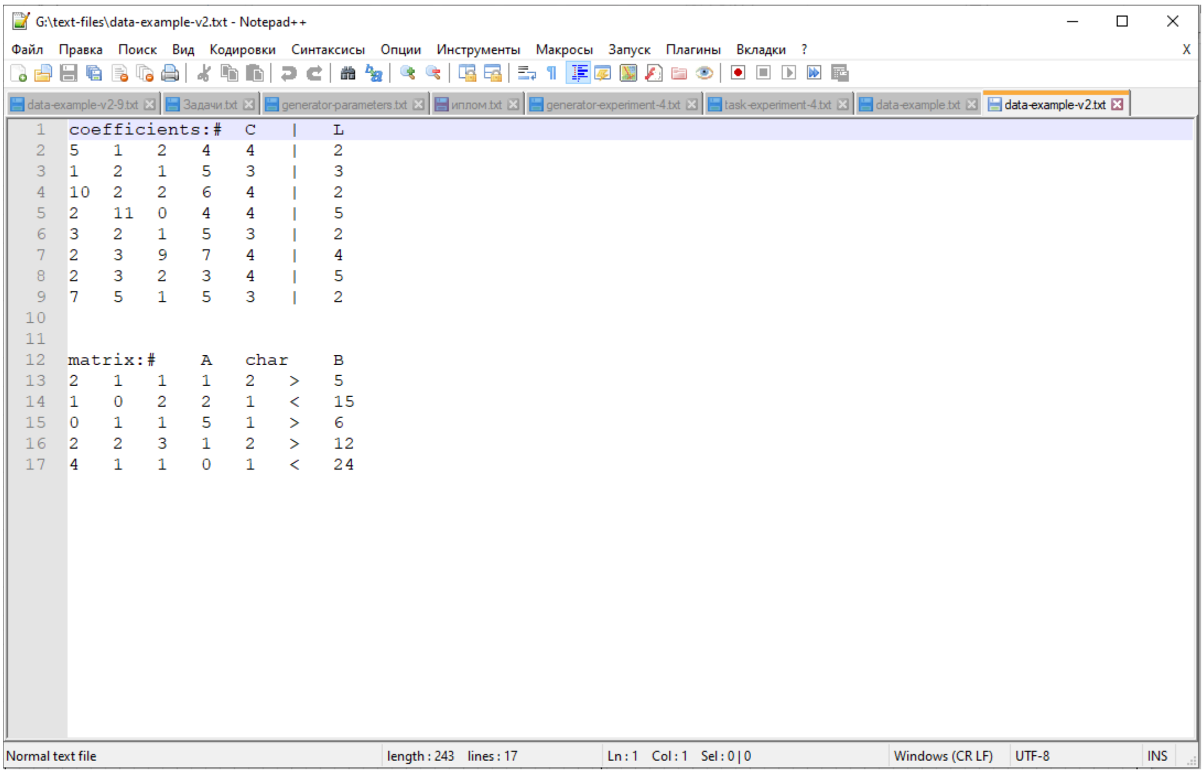
Плакат 3 - Екранна форма головного меню системи після вибору файлу даних, типів експериментів та кількості генерацій для збору статистики



Плакат 4 - Приклад формату файлу даних із параметрами для генератора задач



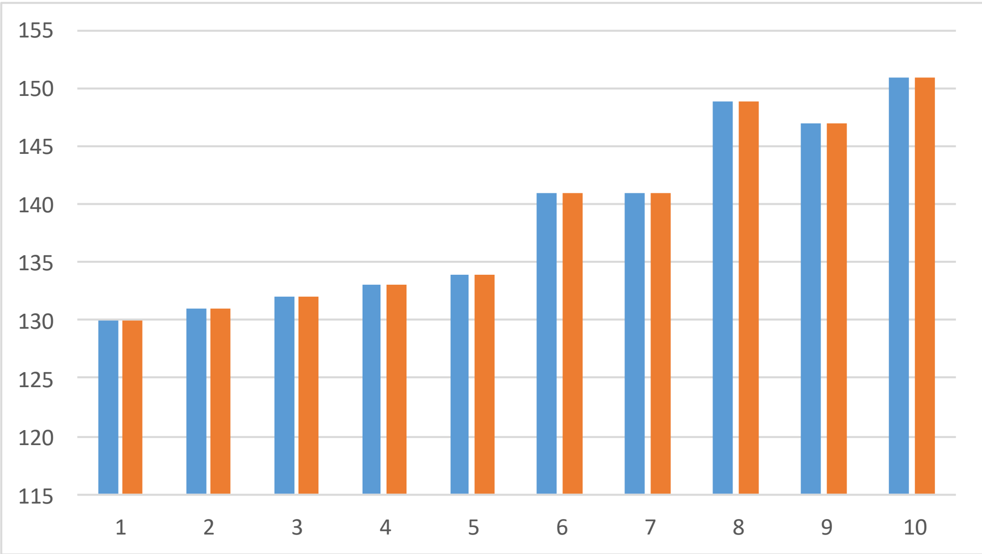
Плакат 5 - Приклад формату файлу даних із задачею лінійного програмування в умовах невизначеності



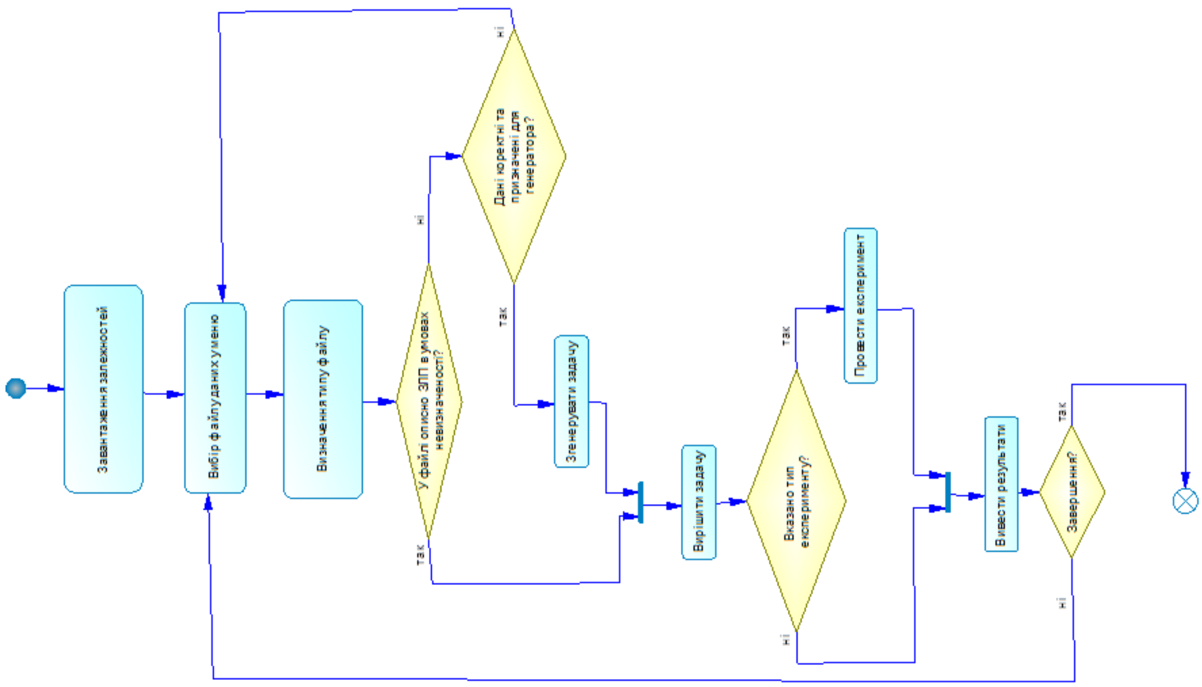
```
coefficients:#  C    |    L
5    1    2    4    4    |    2
1    2    1    5    3    |    3
10   2    2    6    4    |    2
2    11   0    4    4    |    5
3    2    1    5    3    |    2
2    3    9    7    4    |    4
2    3    2    3    4    |    5
7    5    1    5    3    |    2


matrix:#     A    char      B
2    1    1    1    2    >    5
1    0    2    2    1    <    15
0    1    1    5    1    >    6
2    2    3    1    2    >    12
4    1    1    0    1    <    24
```

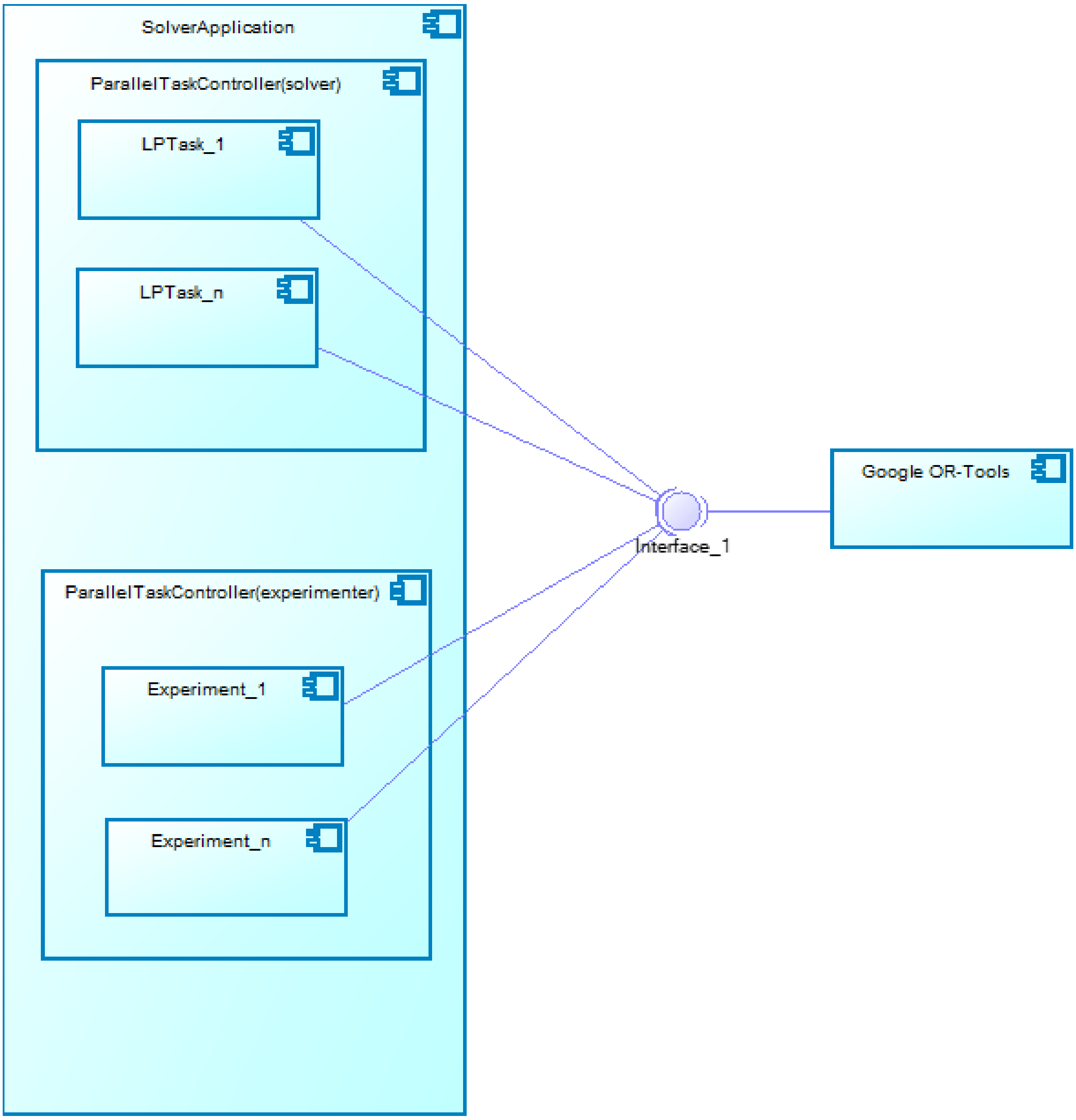
Плакат 6 - Гістограма впливу відсотку похибки у обмеженнях пов'язаних із несуттєвими функціоналами задачі на кількість задач, які змінили розв'язок



Плакат 7 - Схема структурна діяльності системи



Плакат 8 - Схема структурна компонентів системи



Плакат 9 - Схема структурна послідовності діяльності системи